Stability Analysis and Optimal-Control Synthesis via Convex Optimization

Tobia Marcucci

tobiam@mit.edu

Pisa
July 30, 2020

Before starting

- These slides are available at https://github.com/TobiaMarcucci/optimal_control_pisa
- In the same repo you'll find all the Drake demos I show in this presentation
- The buttons ▸ Try this in Drake will open the demos in Google Colab

Introduction

Quick recap

A fairly familiar problem at this point:

$$
\begin{aligned}
v(x_0) := \min_{u,x} \; & \int_0^\infty l(x(t), u(t)) dt \\
\text{subject to } & x(0) = x_0 \\
& \dot{x}(t) = f(x(t), u(t)), && \text{for all } t \in [0, \infty) \\
& u(t) \in U, && \text{for all } t \in [0, \infty)
\end{aligned}
$$

Quick recap

A fairly familiar problem at this point:

$$
\begin{aligned}
v(x_0) := \min_{u,x} \int_0^\infty & l(x(t), u(t)) dt \\
\text{subject to } & x(0) = x_0 \\
& \dot{x}(t) = f(x(t), u(t)), && \text{for all } t \in [0, \infty) \\
& u(t) \in U, && \text{for all } t \in [0, \infty)
\end{aligned}
$$

- Pontryagin's minimum principle (**analytical**):
  - necessary conditions for optimality as a two-point boundary value problem
  - closed-form solution only in very few special cases
- **Numerical** optimization:
  - in general, a nonconvex program
  - convex if system is linear, and objective and constraints are convex

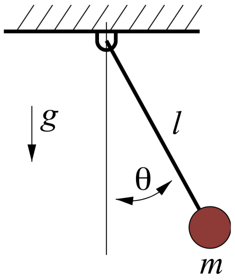
Working with trajectories is not the only option

Classical example: **Lyapunov stability** is much easier in state space than "in time"

Working with trajectories is not the only option

Classical example: **Lyapunov stability** is much easier in state space than "in time"

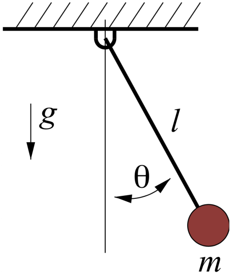
- An ODE such as
$$\ddot{\theta}(t) + \dot{\theta}(t) + \sin(\theta(t)) = 0$$
does not have closed-form solution $\theta(t)$



Working with trajectories is not the only option

Classical example: **Lyapunov stability** is much easier in state space than "in time"

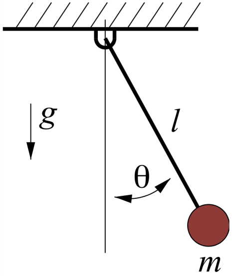
- An ODE such as
$$\ddot{\theta}(t) + \dot{\theta}(t) + \sin(\theta(t)) = 0$$
does not have closed-form solution $\theta(t)$
- Asymptotic stability ($\lim_{t\to\infty} \theta(t) = 0$) can be easily proved by showing that the energy
$$v(\theta, \dot{\theta}) := \frac{1}{2}\dot{\theta}^2 - \cos(\theta)$$
converges to zero



Working with trajectories is not the only option

Classical example: **Lyapunov stability** is much easier in state space than "in time"

- An ODE such as
$$\ddot{\theta}(t) + \dot{\theta}(t) + \sin(\theta(t)) = 0$$
does not have closed-form solution $\theta(t)$
- Asymptotic stability ($\lim_{t\to\infty} \theta(t) = 0$) can be easily proved by showing that the energy
$$v(\theta, \dot{\theta}) := \frac{1}{2}\dot{\theta}^2 - \cos(\theta)$$
converges to zero



Lyapunov theorem (sketch)

$\dot{x} = f(x)$ is stable if there exists $v(x) \geq 0$ such that $\dot{v}(x) = \frac{\partial v}{\partial x}(x) f(x) \leq 0$

Dynamic programming

- The state-space approach to optimal control is called **Dynamic Programming** (DP)
- At its core we have the **Hamilton-Jacobi-Bellman** (HJB) equation

$$\min_{u \in U} \left\{ l(x,u) + \frac{\partial v}{\partial x}(x) f(x,u) \right\} = 0, \quad \text{for all } x$$

recall that $v(x_0)$ is the minimum of the optimal control problem for $x(0) = x_0$

Dynamic programming

- The state-space approach to optimal control is called **Dynamic Programming** (DP)
- At its core we have the **Hamilton-Jacobi-Bellman** (HJB) equation

$$\min_{u \in U} \left\{ l(x,u) + \frac{\partial v}{\partial x}(x) f(x,u) \right\} = 0, \quad \text{for all } x$$

recall that $v(x_0)$ is the minimum of the optimal control problem for $x(0) = x_0$

$\frac{\text{Dynamic programming}}{\text{Optimal control}} = \frac{\text{Lyapunov theorem}}{\text{Stability analisis}}$? Not quite...

- In time, optimal control has similar issues to stability (we end up with hard ODEs)
- In state space:
  - HJB is a nasty nonlinear Partial Differential Equation (PDE)
  - Lyapunov conditions ($v(x) \geq 0$, $\dot{v}(x) \leq 0$) are simple linear differential inequalities

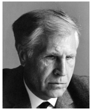
Quoting the authors

Bellman (1957),
"Dynamic Programming"

The problem is not to be considered solved in the mathematical sense until the structure of the optimal policy is understood.

Put another way, in place of determining the optimal sequence of decisions from some *fixed* state of the system, we wish to determine the optimal decision to be made at *any* state of the system. Only if we know the latter, do we understand the intrinsic structure of the solution.

The conceptual advantage of thinking in terms of policies is very great. It affords us a means of thinking about and treating problems which cannot be profitably discussed in any other terms. If we were to hazard a guess as to which direction of research would achieve the greatest success in the future of multi-dimensional processes, we would unhesitatingly choose this one.



Pontryagin et al. (1962),
"The Mathematical Theory of Optimal Processes"

Starting from the assumption that the synthesizing control (12) does exist, and that the corresponding functional (4), which is now a function of the point x:

$$J = J(x) = J(x^1, \cdots, x^n) \qquad (13)$$

is a continuously differentiable function of the variables $x^1, \cdots, x^n$, the American mathematician R. Bellman constructed a partial differential equation for the functional (13). This equation of Bellman's gives rise to another approach to the solution of the optimal control problem (see §9). It is different from the one given in this book, but is closely related to it. It must be noted that the assumption on the continuous differentiability of the functional (13) does not hold in the simplest cases. Thus, Bellman's considerations yield a good heuristic method, rather than a mathematical solution of the problem. The maximum principle, in addition to its complete mathematical validity, also has the advantage that it results in a system of ordinary differential equations, whereas Bellman's approach requires the solution of a partial differential equation.

Our goal today

Use **convex optimization** to design functions in state space:

- SemiDefinite Programming (SDP)
- Sums-of-Squares (SOS) optimization

Our goal today

Use **convex optimization** to design functions in state space:

- SemiDefinite Programming (SDP)
- Sums-of-Squares (SOS) optimization

Two main applications:

- Design of Lyapunov functions
  - Not really optimal control, but of fundamental importance
  - Convex optimization will be a game changer here
- Approximate dynamic programming
  - Convex optimization will be quite effective here

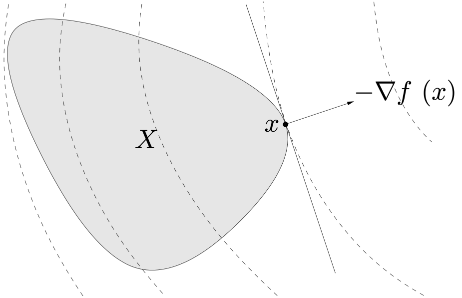
Convex-optimization background

Convex optimization recap

Standard convex program:

$$\min_{x} \; f(x)$$
$$\text{subject to } x \in X$$

- f is a convex function
- X is a convex set



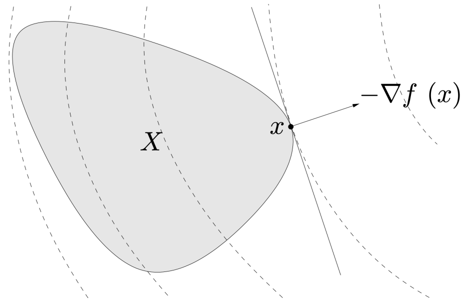
Boyd, Vandenberghe - "Convex Optimization"

Convex optimization recap

Standard convex program:

$$\min_{x} \; f(x)$$
$$\text{subject to } x \in X$$

- f is a convex function
- X is a convex set



Boyd, Vandenberghe - "Convex Optimization"

Why are we so obsessed with convex optimization?

Every local minimum is a **global minimum**

Proof of "every local minimum is global"

- If x^* is a local minimum, there exists $r > 0$ such that

$$x^* = \arg\min_{x \in X} \{f(x) : \|x - x^*\| \leq r\}$$

Proof of "every local minimum is global"

- If x^* is a local minimum, there exists $r > 0$ such that

$$x^* = \arg\min_{x \in X} \{f(x) : \|x - x^*\| \leq r\}$$

- If x^* was not a global minimum, there would exist $y \in X$ such that $f(y) < f(x^*)$
  - and, necessarily, $\|y - x^*\| > r$

Proof of "every local minimum is global"

- If x^* is a local minimum, there exists $r > 0$ such that

$$x^* = \arg\min_{x \in X} \{f(x) : \|x - x^*\| \leq r\}$$

- If x^* was not a global minimum, there would exist $y \in X$ such that $f(y) < f(x^*)$
  - and, necessarily, $\|y - x^*\| > r$
- Take a point z on the line connecting x^* and y, distant $r/2$ from x^*:

$$z := x^* + \theta(y - x^*) = (1 - \theta)x^* + \theta y, \quad \theta := \frac{r}{2\|y - x^*\|}$$

Proof of "every local minimum is global"

- If x^* is a local minimum, there exists $r > 0$ such that

$$x^* = \arg\min_{x \in X} \{f(x) : \|x - x^*\| \leq r\}$$

- If x^* was not a global minimum, there would exist $y \in X$ such that $f(y) < f(x^*)$
  - and, necessarily, $\|y - x^*\| > r$
- Take a point z on the line connecting x^* and y, distant $r/2$ from x^*:

$$z := x^* + \theta(y - x^*) = (1 - \theta)x^* + \theta y, \quad \theta := \frac{r}{2\|y - x^*\|}$$

- By convexity of the feasible set X, $z \in X$

Proof of "every local minimum is global"

- If x^* is a local minimum, there exists $r > 0$ such that

$$x^* = \arg\min_{x \in X} \{f(x) : \|x - x^*\| \leq r\}$$

- If x^* was not a global minimum, there would exist $y \in X$ such that $f(y) < f(x^*)$
  - and, necessarily, $\|y - x^*\| > r$
- Take a point z on the line connecting x^* and y, distant $r/2$ from x^*:

$$z := x^* + \theta(y - x^*) = (1 - \theta)x^* + \theta y, \quad \theta := \frac{r}{2\|y - x^*\|}$$

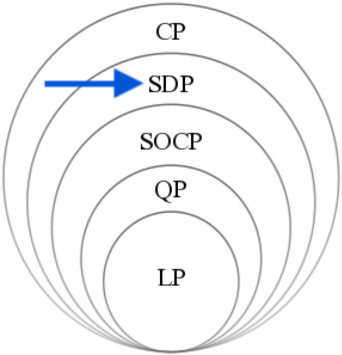
- By convexity of the feasible set X, $z \in X$
- By convexity of the objective,

$$f(z) \leq (1 - \theta)f(x^*) + \theta f(y) < (1 - \theta)f(x^*) + \theta f(x^*) = f_0(x^*)$$

which contradicts the local optimality of x^*

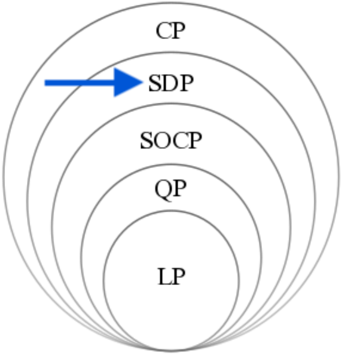
A hierarchy of Convex Programs (CPs)

- Linear Programs (LPs) are the easiest
- SemiDefinite Programming (SDP) is a broad class of CPs that can be solved efficiently (our main tool today)
- Some CPs outside the class of SDP can be quite hard



A hierarchy of Convex Programs (CPs)

- Linear Programs (LPs) are the easiest
- SemiDefinite Programming (SDP) is a broad class of CPs that can be solved efficiently (our main tool today)
- Some CPs outside the class of SDP can be quite hard



What? Hard convex optimizations??

- Optimization algorithms require to check how far a point is from being infeasible
- Some convex sets are very hard to describe mathematically!
- We'll come back to this point...

Definite symmetric matrices

Definition

Equivalent definitions for a symmetric matrix A to be **positive semidefinite** ($A \succeq 0$):

- $x^T A x \geq 0$ for all x
- The eigenvalues $\lambda_1, \ldots, \lambda_n$ of A are nonnegative
- There exists L such that $A = L^T L$

Definite symmetric matrices

Definition

Equivalent definitions for a symmetric matrix A to be **positive semidefinite** ($A \succeq 0$):

- $x^T A x \geq 0$ for all x
- The eigenvalues $\lambda_1, \ldots, \lambda_n$ of A are nonnegative
- There exists L such that $A = L^T L$

Why do we care about positive semidefinite matrices?

- The set $\mathbb{S}_n^+ := \{(a_{11}, \ldots, a_{nn}) : A \succeq 0\}$ is convex, where

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{12} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{nn} \end{bmatrix}$$

Proof of "$\mathbb{S}_n^+$ is convex"

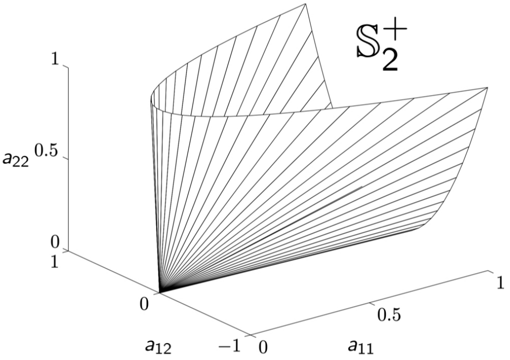
Just use the definition:

- Assume $A_1 \succeq 0$ and $A_2 \succeq 0$
- Take a convex combination of A_1 and A_2:

$$A := \theta A_1 + (1-\theta)A_2, \quad \theta \in [0,1]$$

- Then, for all x,

$$x^T A x = \theta x^T A_1 x + (1-\theta) x^T A_2 x \geq 0$$



Boyd, Vandenberghe - "Convex Optimization"

Semidefinite program in standard form

$$
\begin{aligned}
\min_{X}\ & \operatorname{tr}(CX) \\
\text{subject to}\ & \operatorname{tr}(A_i X) = b_i, \quad i = 1, \ldots, p \\
& X \succeq 0
\end{aligned}
$$

- recall that $\operatorname{tr}(CX) = \sum_{i,j=1}^{n} C_{ij} x_{ij}$, i.e., an arbitrary linear function of the entries of A
- linear objective function (convex)
- p linear equality constraints (convex)
- one semideifinite constraint (convex)

Semidefinite program in standard form

$$\begin{aligned} \min_{X} \ & \operatorname{tr}(CX) \\ \text{subject to } & \operatorname{tr}(A_i X) = b_i, \quad i = 1, \ldots, p \\ & X \succeq 0 \end{aligned}$$

- recall that $\operatorname{tr}(CX) = \sum_{i,j=1}^{n} C_{ij} x_{ij}$, i.e., an arbitrary linear function of the entries of A
- linear objective function (convex)
- p linear equality constraints (convex)
- one semideifinite constraint (convex)

Why can we solve SDPs efficiently?

- We need a function to tell how far is a point $(a_{11}, \ldots, a_{nn})$ from being infeasible
- Interior of the feasible set is $\{(a_{11}, \ldots, a_{nn}) : \lambda_1 > 0, \ldots, \lambda_n > 0\}$
- Natural candidate is

$$-\sum_{i=1}^{n} \ln(\lambda_i) = -\ln\left(\prod_{i=1}^{n} \lambda_i\right) = -\ln(\det A)$$

Sums-of-squares optimization

Nonlinear parameterization of the function space

- Ultimately, we want to use convex optimization to design functions $v(x) : \mathbb{R}^n \mapsto \mathbb{R}$
- **First step**: parameterize $v(x)$ with a finite number of coefficients (optimization variables)

Nonlinear parameterization of the function space

- Ultimately, we want to use convex optimization to design functions $v(x) : \mathbb{R}^n \mapsto \mathbb{R}$
- **First step**: parameterize $v(x)$ with a finite number of coefficients (optimization variables)

Nonnlinear parameterization

$$v(x) := \psi(x, \alpha)$$

- $\alpha \in \mathbb{R}^r$ is the vector of coefficients
- ψ is a given nonlinear scalar function

Nonlinear parameterization of the function space

- Ultimately, we want to use convex optimization to design functions $v(x) : \mathbb{R}^n \mapsto \mathbb{R}$
- **First step**: parameterize $v(x)$ with a finite number of coefficients (optimization variables)

Nonlinear parameterization

$$v(x) := \psi(x, \alpha)$$

- $\alpha \in \mathbb{R}^r$ is the vector of coefficients
- ψ is a given nonlinear scalar function

Say we want $v(0) = 0$:

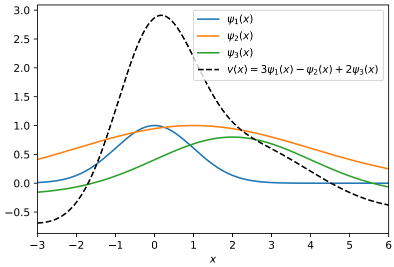
- $\psi(0, \alpha) = 0$ is a **nonlinear equality constraint** in α (not convex!)

Linear parameterization

Linear parameterization

$$v(x) := \alpha^T \psi(x)$$

- $\alpha \in \mathbb{R}^r$ is the vector of coefficients
- ψ is an r-vector of the basis functions

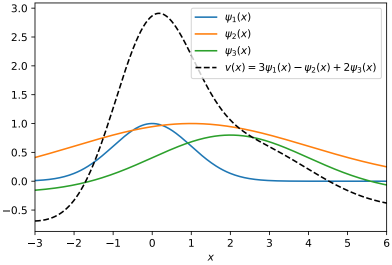


Linear parameterization

Linear parameterization

$$v(x) := \alpha^T \psi(x)$$

- $\alpha \in \mathbb{R}^r$ is the vector of coefficients
- ψ is an r-vector of the basis functions



Let's try again $v(0) = 0$:

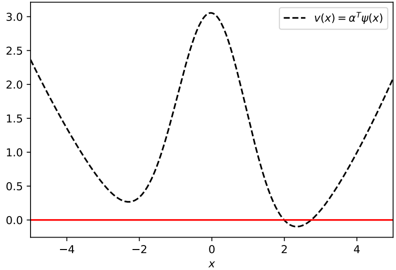
- $\alpha^T \psi(0) = 0$ is a **linear equality constraint** in α (convex)

Other ways we might want to constrain $v(x)$?

Nonnegativity constraints

For example, a Lyapunov function must be nonnegative

$$v(x) := \alpha^T \psi(x) \geq 0 \text{ for all } x$$

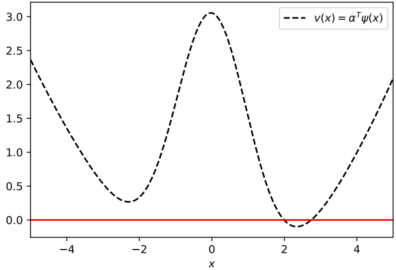


Other ways we might want to constrain $v(x)$?

Nonnegativity constraints

For example, a Lyapunov function must be nonnegative

$$v(x) := \alpha^T \psi(x) \geq 0 \text{ for all } x$$



Let's analyze the set

$$\{\alpha : \alpha^T \psi(x) \geq 0 \text{ for all } x\} \subseteq \mathbb{R}^r$$

Nonnegativity constraints and linear parameterizations

The set $\{\alpha : \alpha^T \psi(x) \geq 0\}$ is convex

- Assume α_1 is such that $v_1(x) := \alpha_1^T \psi(x) \geq 0$, and the same for α_2
- Take a convex combination of α_1 and α_2

$$\alpha := \theta \alpha_1 + (1 - \theta)\alpha_2, \quad \theta \in [0, 1]$$

- Then $v(x) := \alpha^T \psi(x) = \theta v_1(x) + (1 - \theta) v_2(x) \geq 0$

Nonnegativity constraints and linear parameterizations

The set $\{\alpha : \alpha^T \psi(x) \geq 0\}$ is convex

- Assume α_1 is such that $v_1(x) := \alpha_1^T \psi(x) \geq 0$, and the same for α_2
- Take a convex combination of α_1 and α_2

$$\alpha := \theta\alpha_1 + (1-\theta)\alpha_2, \quad \theta \in [0,1]$$

- Then $v(x) := \alpha^T \psi(x) = \theta v_1(x) + (1-\theta) v_2(x) \geq 0$

Does this mean that optimizing over nonnegative $v(x)$ is easy?

- No, in general, no
- Except for a few cases, for a given α, determining if $\alpha^T \psi(x) \geq 0$ is NP-hard!

Back to the black board

What about a parameterization that is nonnegative by construction?

Back to the black board

What about a parameterization that is nonnegative by construction?

Sums Of Squares (SOS)

$$v(x) := \psi^T(x) Q \psi(x), \quad Q \succeq 0$$

- Q is an r-by-r matrix of coefficients (entries of Q are our optimization variables)
- ψ is an r-vector of basis functions

Back to the black board

What about a parameterization that is nonnegative by construction?

Sums Of Squares (SOS)

$$v(x) := \psi^T(x) Q \psi(x), \quad Q \succeq 0$$

- Q is an r-by-r matrix of coefficients (entries of Q are our optimization variables)
- ψ is an r-vector of basis functions

Remarks:

- $v(x) = \sum_{i,j=1}^{r} Q_{ij} \psi_i(x) \psi_j(x)$ is still linear in Q
    - E.g., $v(0) = 0$ is a **linear constraint** on the entries of Q
- $Q \succeq 0$ is a **convex constraint**

Use SOS as a constraints

What if we want to certify that a given function $v(x)$ is nonnegative?

Use SOS as a constraints

What if we want to certify that a given function $v(x)$ is nonnegative?

- We can try to find a SOS decomposition $v(x) = \psi^T(x) Q \psi(x)$, with $Q \succeq 0$
- The issue is how to pick $\psi(x)$...

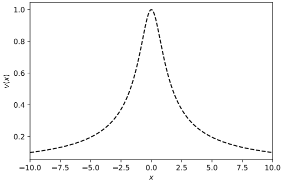
Use SOS as a constraints

What if we want to certify that a given function $v(x)$ is nonnegative?

- We can try to find a SOS decomposition $v(x) = \psi^T(x) Q \psi(x)$, with $Q \succeq 0$
- The issue is how to pick $\psi(x)$...

Example

- Consider $v(x) = \frac{1}{\sqrt{1+x^2}}$
- Nonnegative for all x
- I don't quite know how to pick $\psi(x)$ here



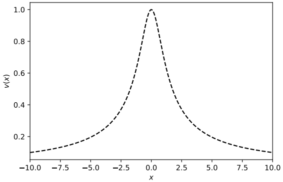
Use SOS as a constraints

What if we want to certify that a given function $v(x)$ is nonnegative?

- We can try to find a SOS decomposition $v(x) = \psi^T(x) Q \psi(x)$, with $Q \succeq 0$
- The issue is how to pick $\psi(x)$...

Example

- Consider $v(x) = \frac{1}{\sqrt{1+x^2}}$
- Nonnegative for all x
- I don't quite know how to pick $\psi(x)$ here



Can you think of any class of functions for which this approach might work?

SOS polynomials

Example

- Given $v(x) = 2 - 2x + 3x^2 + 2x^3 + x^4$, we want $\psi(x)$ such that $v(x) = \psi^T(x)Q\psi(x)$
- We can just pick $\psi(x) := (1, x, x^2)$
- Then, we look for $Q \succeq 0$ such that

$$2 - 2x + 3x^2 + 2x^3 + x^4 = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{12} & Q_{22} & Q_{23} \\ Q_{13} & Q_{23} & Q_{33} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$$

- Comparing the coefficients we get **linear constraints** on Q, e.g., $Q_{11} = 2$

SOS polynomials

Example

- Given $v(x) = 2 - 2x + 3x^2 + 2x^3 + x^4$, we want $\psi(x)$ such that $v(x) = \psi^T(x) Q \psi(x)$
- We can just pick $\psi(x) := (1, x, x^2)$
- Then, we look for $Q \succeq 0$ such that

$$2 - 2x + 3x^2 + 2x^3 + x^4 = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{12} & Q_{22} & Q_{23} \\ Q_{13} & Q_{23} & Q_{33} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$$

- Comparing the coefficients we get **linear constraints** on Q, e.g., $Q_{11} = 2$

- For $v(x)$ **polynomial** of degree $2d$, we fill $\psi(x)$ with all the **monomials** up to degree d

SOS polynomials

Example

- Given $v(x) = 2 - 2x + 3x^2 + 2x^3 + x^4$, we want $\psi(x)$ such that $v(x) = \psi^T(x) Q \psi(x)$
- We can just pick $\psi(x) := (1, x, x^2)$
- Then, we look for $Q \succeq 0$ such that

$$2 - 2x + 3x^2 + 2x^3 + x^4 = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{12} & Q_{22} & Q_{23} \\ Q_{13} & Q_{23} & Q_{33} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$$

- Comparing the coefficients we get **linear constraints** on Q, e.g., $Q_{11} = 2$

- For $v(x)$ **polynomial** of degree $2d$, we fill $\psi(x)$ with all the **monomials** up to degree d
- This works even if the coefficients of v are (linear functions of) optimization variables

SOS polynomials

Example

- Given $v(x) = 2 - 2x + 3x^2 + 2x^3 + x^4$, we want $\psi(x)$ such that $v(x) = \psi^T(x) Q \psi(x)$
- We can just pick $\psi(x) := (1, x, x^2)$
- Then, we look for $Q \succeq 0$ such that

$$2 - 2x + 3x^2 + 2x^3 + x^4 = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T \begin{bmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{12} & Q_{22} & Q_{23} \\ Q_{13} & Q_{23} & Q_{33} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$$

- Comparing the coefficients we get **linear constraints** on Q, e.g., $Q_{11} = 2$

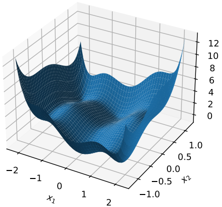
- For $v(x)$ **polynomial** of degree $2d$, we fill $\psi(x)$ with all the **monomials** up to degree d
- This works even if the coefficients of v are (linear functions of) optimization variables
- Other classes of functions could work, polynomials have many properties

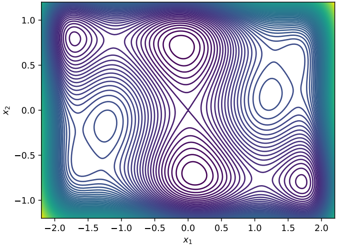
The power of SOS optimization

Try this in Drake

Minimize the six-hump-camel function

$$\min_{x} p(x) = 4x_1^2 + x_1x_2 - 4x_2^2 - 2.1x_1^4 + 4x_2^4 + x_1^6/3$$





The power of SOS optimization

▸ Try this in Drake

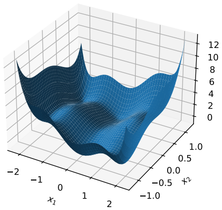
Minimize the six-hump-camel function

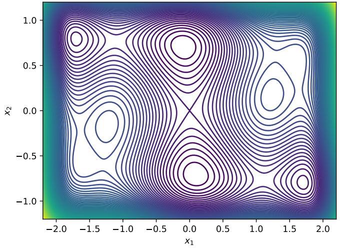
$$\min_x p(x) = 4x_1^2 + x_1x_2 - 4x_2^2 - 2.1x_1^4 + 4x_2^4 + x_1^6/3$$

- Write it as

$$\max_{\lambda,Q} \lambda$$
$$\text{subject to } p(x) - \lambda \text{ is SOS}$$

where “is SOS” means “equal to a SOS polynomial in x parameterized by Q”





Did we just solve global optimization over polynomials?

Not all the nonnegative polynomials are SOS!

- Hilbert in 1888[1]showed that "SOS = nonnegative" only in 3 cases[2]:
  - Univariate polynomials (1 variable)
  - Quadratic polynomials (degree 2)
  - Bivariate quartics (2 variables, degree 4)

[1] Hilbert - "Ueber die Darstellung definiter Formen als Summe von Formenquadraten"
[2] See also Hilbert's 17th problem

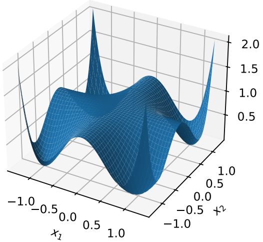
Did we just solve global optimization over polynomials?

Not all the nonnegative polynomials are SOS!

- Hilbert in 1888[1] showed that "SOS $=$ nonnegative" only in 3 cases[2]:
  - Univariate polynomials (1 variable)
  - Quadratic polynomials (degree 2)
  - Bivariate quartics (2 variables, degree 4)
- Famous counterexample by Motzkin

$$x_1^4 x_2^2 + x_1^2 x_2^4 + 1 - 3x_1^2 x_2^2$$

- Counterexamples are "rare enough that people give names to them"



[1] Hilbert - "Ueber die Darstellung definiter Formen als Summe von Formenquadraten"
[2] See also Hilbert's 17th problem

One more trick: S-procedure for constrained SOS

- To constrain the polynomial $v(x)$ to be nonnegative over the set

$$\{x : g(x) = 0\}$$

with $g(x)$ polynomial, we can add a new polynomial $\lambda(x)$ and write

$$v(x) + \lambda(x)g(x) \text{ is SOS}$$

One more trick: S-procedure for constrained SOS

- To constrain the polynomial $v(x)$ to be nonnegative over the set

$$\{x : g(x) = 0\}$$

with $g(x)$ polynomial, we can add a new polynomial $\lambda(x)$ and write

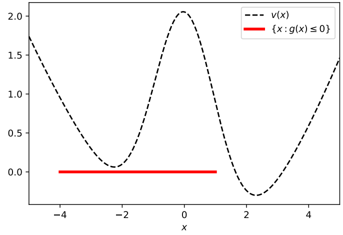
$$v(x) + \lambda(x)g(x) \text{ is SOS}$$

- Similarly, for the set

$$\{x : g(x) \leq 0\}$$

we can add a new SOS polynomial $\lambda(x)$ and write

$$v(x) + \lambda(x)g(x) \text{ is SOS}$$



Design of Lyapunov functions

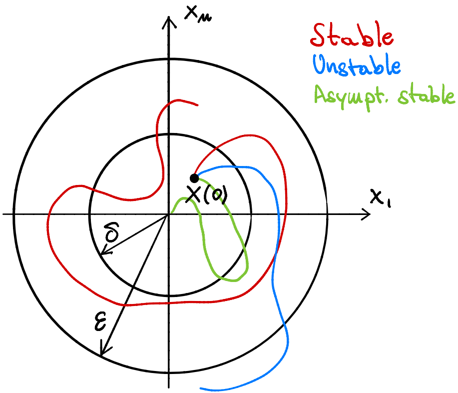
Recap on Lyapunov stability

The equilibrium $x = 0$ for the system $\dot{x} = f(x)$ is:

- **stable** if, for all $\varepsilon > 0$, there exists $\delta > 0$ such that

$$\|x(0)\| < \delta \Rightarrow \|x(t)\| < \varepsilon \text{ for all } t \geq 0$$

- **asymptotically stable** if it is stable, and $\lim_{t\to\infty} \|x(t)\| = 0$



Recap on Lyapunov direct method

The equilibrium $x = 0$ for the system $\dot{x} = f(x)$ is:

- **stable** iff there exists $v(x)$ such that

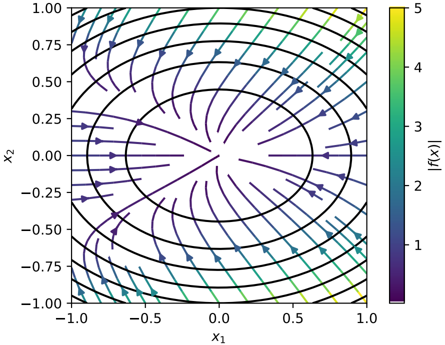
$$v(0) = 0, \quad v(x) > 0 \text{ for all } x \neq 0,$$

  and

$$\dot{v}(x) = \frac{\partial v}{\partial x}(x) f(x) \leq 0 \text{ for all } x \neq 0$$

- **asymptotically stable** iff

$$\dot{v}(x) < 0 \text{ for all } x \neq 0$$



Design of Lyapunov functions via SOS

SOS parameterization of the Lyapunov function

$$v(x) = \psi^T(x) Q \psi(x), \quad Q \succeq 0$$

Filling $\psi(x)$ with monomials is the natural choice:

- It works in the linear case $f(x) = Ax$ (we always have $v(x) = x^T P x$)
- If $f(x)$ is polynomial, then

  $$\dot{v}(x) = \frac{\partial v}{\partial x}(x) f(x)$$

  is also polynomial, and its coefficients are linear in Q
- If $f(x)$ is not polynomial we can use, e.g., Taylor approximation

Design of Lyapunov functions via SOS

Try this in Drake

The overall SOS program

$$\text{find } v(x) \text{ SOS} : \frac{\partial v}{\partial x}(x) f(x) \text{ is SOS}$$

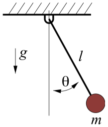
Miscellaneous observations:

- To avoid the trivial solution $v(x) = 0$ we can add a linear constraint like $v(1) = 1$
- Be aware: not all the stable polynomial systems admit a polynomial Lyapunov function[3]

[3] Ahmadi, Krstic, Parrilo - "A Globally Asymptotically Stable Polynomial Vector Field with no Polynomial Lyapunov Function"

Verification of nonpolynomial systems via SOS

▸ Try this in Drake



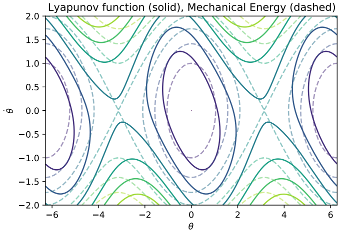
- Some non-polynomial systems can be verified exactly using SOS[4]

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -x_2 - \sin(x_1)$$

- Introduce auxiliary variables $s = \sin(x_1)$, $c = \cos(x_1)$
- Substitute to get the polynomial system

$$\dot{s} = cx_2, \quad \dot{c} = -sx_2, \quad \dot{x}_2 = -x_2 - s$$

subject to the polynomial constraint $s^2 + c^2 = 1$



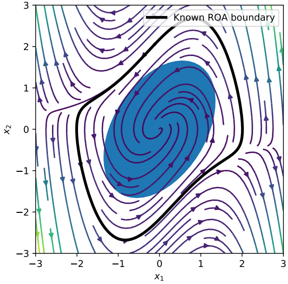
[1]See also Papachristodoulou, Prajna - "Analysis of Non-polynomial Systems using the Sum of Squares Decomposition"

Bonus: approximation of region of attraction

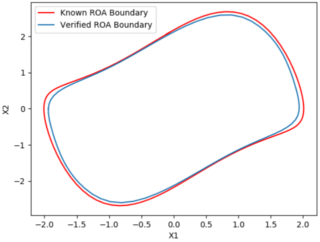
Time-reversed Van der Pol oscillator

$$\dot{x}_1 = -x_2$$
$$\dot{x}_2 = x_1 + (x_1^2 - 1)x_2$$

Maximize volume of level set



Alternate between maximization of level set and reshape Lyapunov function



Control-Lypaunov function

Can use use SOS + Lyapunov verification to design controllers? Unfortunately no...

- We let $v(x)$ be a SOS polynomial
- We use a linear parameterization for $u(x)$ (polynomial)
- For simplicity, let the system be polynomial and control-affine

$$\dot{x} = f(x) + G(x)u$$

- Lyapunov condition becomes

$$-\frac{\partial v}{\partial x}(x)[f(x) + G(x)u(x)] \text{ is SOS}$$

- Multiplication of $\frac{\partial v}{\partial x}$ and u leads to **nonlinear equality constraints**

Approximate dynamic programming

Hamilton-Jacobi-Bellman (HJB) equation

How to arrive to the HJB equation

$$\min_{u \in U} \left\{ l(x,u) + \frac{\partial v}{\partial x}(x) f(x,u) \right\} = 0, \quad \text{for all } x$$

from the Optimal Control Problem (OCP)

$$\begin{aligned} v(x_0) := \min_{u,x} \; & \int_0^\infty l(x(t), u(t)) dt \\ \text{subject to } & x(0) = x_0 \\ & \dot{x}(t) = f(x(t), u(t)), && \text{for all } t \in [0, \infty) \\ & u(t) \in U, && \text{for all } t \in [0, \infty) \end{aligned}$$

HJB: informal derivation

First order approximation of the control problem with time step h:

- Cost function

$$\int_0^{\infty} l(x(t), u(t))dt \approx h \sum_{t=0}^{\infty} l(x_t, u_t)$$

- Dynamics

$$\dot{x}(t) = f(x(t), u(t)) \Rightarrow x_{t+1} = x_t + hf(x_t, u_t)$$

- Overall

$$v(x_0) := \min_{u,x} h \sum_{t=0}^{\infty} l(x_t, u_t)$$

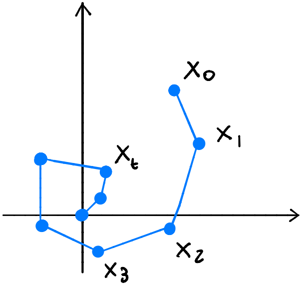
$$\begin{aligned} \text{subject to } & x_0 \text{ given} \\ & x_{t+1} = x_t + hf(x_t, u_t), & t = 0, \dots, \infty \\ & u_t \in U, & t = 0, \dots, \infty \end{aligned}$$

HJB: informal derivation

The dynamic-programming principle:

- Assume $\mathbf{x}_0 = (x_0, x_1, x_2, x_3, \ldots)$ is the optimal trajectory from x_0 to the origin
- Because of the infinite horizon, the optimal trajectory $\mathbf{x}_1$ from x_1 to the origin must be $(x_1, x_2, x_3, \ldots)$
  - Otherwise $(x_0, \mathbf{x}_1)$ would be better than $\mathbf{x}_0$ (contradiction)
- If we were to know the cost-to-go $v(x_1)$ from x_1, then we could just solve a one-step problem

$$\begin{aligned} v(x_0) &= \min_{u_0 \in U} \{hl(x_0, u_0) + v(x_1)\} \\ &= \min_{u_0 \in U} \{hl(x_0, u_0) + v(x_0 + hf(x_0, u_0))\} \end{aligned}$$



HJB: informal derivation

- If h is very small,

$$v(x_0 + hf(x_0, u_0)) \approx v(x_0) + h\frac{\partial v}{\partial x}(x_0)f(x_0, u_0)$$

- Substituting

$$v(x_0) = \min_{u_0 \in U} \left\{ hl(x_0, u_0) + v(x_0) + h\frac{\partial v}{\partial x}(x_0)f(x_0, u_0) \right\}$$

- Simplifying $v(x_0)$ and dividing by h

$$0 = \min_{u_0 \in U} \left\{ l(x_0, u_0) + \frac{\partial v}{\partial x}(x_0)f(x_0, u_0) \right\}$$

Observations:

- If we know the value function $v(x)$, the optimal controller is the "argmin" of the HJB
- Why "informal"? Where does this derivation leak? (Remember the bang bang controller)

Solving the HJB

The HJB equation is a nonlinear Partial Differential Equation (PDE)

- Analytic solutions only in a few special cases (LQR)
- "Exact" numerical solution is very hard (discretization of the state and control space)
- Can we still find approximate solutions that are "good enough"?

"One side of the HJB equation is convex"

$$\min_{u \in U} \left\{ l(x, u) + \frac{\partial v}{\partial x}(x) f(x, u) \right\} \geq 0, \quad \text{for all } x$$

is equivalent to the **Bellman inequality**

$$l(x, u) + \frac{\partial v}{\partial x}(x) f(x, u) \geq 0, \quad \text{for all } x \text{ and } u \in U$$

- We can relax this condition via SOS!

Bellman inequality

What do we loose by enforcing only one side of the HJB equations?

- Let $x(t)$ and $u(t)$ be the optimal trajectory and control from $x(0)$
- Integrate the Bellman inequality along the optimal trajectory

$$
\begin{aligned}
0 &\leq \int_0^\infty \left[l(x(t), u(t)) + \frac{\partial v}{\partial x}(x(t)) f(x(t), u(t)) \right] dt \\
&= \int_0^\infty l(x(t), u(t)) dt + \int_0^\infty \dot{v}(x(t)) dt \\
&= \int_0^\infty l(x(t), u(t)) dt + v(x(\infty)) - v(x(0))
\end{aligned}
$$

- Let $v(0) = 0$, then $v(x)$ lower bounds the optimal value function

$$
v(x(0)) \leq \int_0^\infty l(x(t), u(t)) dt
$$

SOS for approximate dynamic programming

$$\max_{v} \int_X v(x)dx$$
$$\text{subject to } l(x,u) + \frac{\partial v}{\partial x}(x) f(x,u) \text{ is SOS for all } x \text{ and } u \in U$$
$$v(0) = 0$$

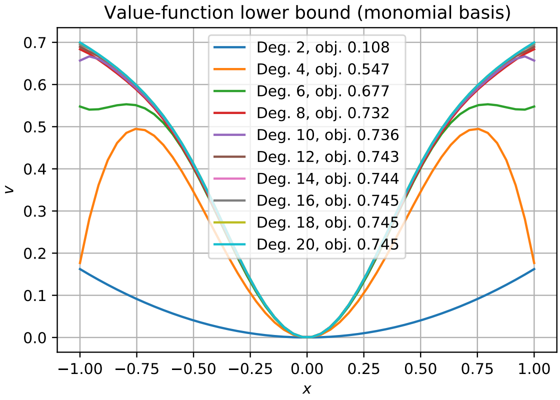
Interpretation:

- Constraints ensure that $v(x)$ lower bounds the value function
- Objective "pushes up" $v(x)$ over the set X
- As the degree of $v(x)$ increases, we get a better and better approximation of the value function over X

Toy approximate-DP problem

Try this in Drake

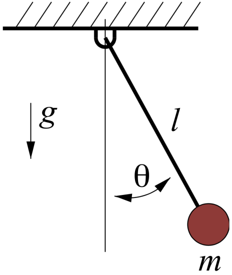
- Scalar dynamics $f(x, u) = x - 4x^3 + u$
- Quadratic running cost $l(x, u) = x^2 + u^2$
- Control limits $u \in U = [-1, 1]$
- Approximate the value function for $x \in X = [-1, 1]$



Pendulum swing up

▸ Try this in Drake

- Find a controller that stabilizes the pendulum in the upright configuration
- Use S-procedure to handle sines and cosines
- Approximate value function of degree 10



Bibliography

Software:

- Drake https://drake.mit.edu

Convex optimization:

- Boyd, Vandenberghe - "Convex Optimization"

Semidefinite programming:

- Vandenberghe, Boyd - "Semidefinite programming"
- Boyd, Vandenberghe - "Convex Optimization"

Sums-of-Squares optimization:

- Parrilo - "Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization"
- Blekherman, Parrilo, Thomas - "Semidefinite Optimization and Convex Algebraic Geometry"

Stability theory:

- Khalil - "Nonlinear Systems"
- Slotine, Li - "Applied nonlinear control"

Bibliography

Dynamic programming:

- Bertsekas - "Dynamic Programming and Optimal Control, Volume 1"
- Kirk - "Optimal Control Theory an Introduction"
- Bressan, Piccoli - "Introduction to the Mathematical Theory of Control"

Hamilton-Jacobi-Bellman equation:

- Evans - "Partial Differential Equations"

Approximate dynamic programming:

- Bertsekas - "Dynamic Programming and Optimal Control, Volume 2"
- De Farias, Van Roy - "The linear programming approach to approximate dynamic programming"
- Lasserre et al. - "Nonlinear optimal control via occupation measures and LMI-relaxations"
- Wang, Boyd - "Approximate dynamic programming via iterated Bellman inequalities"
- Other works from Lygeros' group, e.g. Summers et al. - "Approximate Dynamic Programming via Sum of Squares Programming"