# Underactuated Robotics

***Algorithms for Walking, Running, Swimming, Flying, and Manipulation***

Russ Tedrake


How to cite these notes

**Note:** These are working notes used for a course being taught at MIT. They will be updated throughout the Spring 2019 semester. Lecture videos are available on YouTube.

Previous chapter Table of contents Next chapter

## Chapter 10
## Lyapunov Analysis

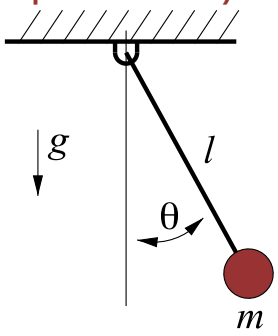
Optimal control provides a powerful framework for formulating control problems using the language of optimization. But solving optimal control problems for nonlinear systems is hard! In many cases, we don't really care about finding the *optimal* controller, but would be satisfied with any controller that is guaranteed to accomplish the specified task. In many cases, we still formulate these problems using computational tools from optimization, and in this chapter we'll learn about tools that can provide guaranteed control solutions for systems that are beyond the complexity for which we can find the optimal feedback.

There are many excellent books on Lyapunov analysis; for instance [10] is an excellent and very readable reference and [61] can provide a rigorous treatment. In this chapter I will summarize (without proof) some of the key theorems from Lyapunov analysis, but then will also introduce a number of numerical algorithms... many of which are new enough that they have not yet appeared in any mainstream textbooks.

### 10.1 Lyapunov Functions

Let's start with our favorite simple example.

**Example 10.1 (Stability of the Damped Pendulum)**




Recall that the equations of motion of the damped simple pendulum are given by

$$ml^2\ddot{\theta} + mgl\sin\theta = -b\dot{\theta},$$

which I've written with the damping on the right-hand side to remind us that it is an external torque that we've modeled.

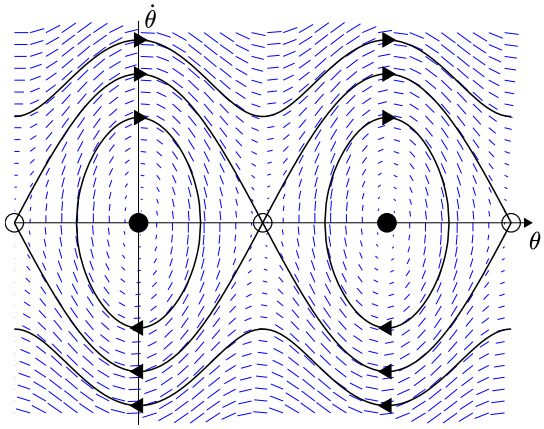
These equations represent a simple second-order differential equation; in chapter 2 we discussed at some length what was known about the solutions to this differential equation--even without damping the equation for $\theta(t)$ as a function of $\theta(0)$, $\dot{\theta}(0)$, involves elliptic integrals of the first kind, and with damping we don't even have that. Since we couldn't provide a solution

analytically, in chapter 2 we resorted to a graphical analysis, and confirmed the intuition that there are fixed points in the system (at $\theta = k\pi$ for every integer $k$) and that the fixed points at $\theta = 2\pi k$ are asymptotically stable with a large basin of attraction. The graphical analysis gave us this intuition, but can we actually prove this stability property? In a way that might also work for much more complicated systems?

One route forward was from looking at the total system energy (kinetic + potential), which we can write down:

$$E(\theta, \dot{\theta}) = \frac{1}{2} m l^2 \dot{\theta}^2 - mgl \cos\theta.$$

Recall that the contours of this energy function are the orbits of the undamped pendulum.




A natural route to proving the stability of the downward fixed points is by arguing that energy decreases for the damped pendulum (with $b > 0$) and so the system will eventually come to rest at the minimum energy, $E = -mgl$, which happens at $\theta = 2\pi k$. Let's make that argument slightly more precise.

Evaluating the time derivative of the energy reveals

$$\frac{d}{dt} E = -b\dot{\theta}^2 \leq 0.$$

This is sufficient to demonstrate that the energy will never increase, but it doesn't actually prove that the energy will converge to the minimum. To take that extra step, we must observe that for most of the states with $\dot{E} = 0$, which for $b > 0$ is the manifold where $\dot{\theta} = 0$, the system can't actually remain in that state, but will rather pass through and enter another region where it continues to lose energy. In fact, the fixed points are the only subset of this $\dot{E} = 0$ manifold where the system can stay on the manifold. Therefore, unless the system is at a fixed point, it will continue to dissipate energy. And therefore we can conclude that as $t \rightarrow \infty$, the system will indeed come to rest at a fixed point (though it could be any fixed point with an energy less than or equal to the initial energy in the system, $E(0)$).

This is an important example. It demonstrated that we could use a relatively simple function, the system energy, to describe something about the long-term dynamics of the pendulum even though the actual trajectories of the system are (analytically) very complex. It also demonstrated one of the subtleties of using an energy-like function that is non-increasing (instead of strictly decreasing) to prove asymptotic stability.

Lyapunov functions generalize this notion of an energy function to more general systems, which might not be stable in the sense of some mechanical energy. If I can find any positive function, call it $V(\mathbf{x})$, that gets smaller over time as the system evolves, then I can potentially use $V$ to make a statement about the long-term behavior of the system. $V$ is called a *Lyapunov function*.

Recall that we defined three separate notions for stability of a fixed-point of a nonlinear system: stability i.s.L., asymptotic stability, and exponential stability. We can use Lyapunov functions to

demonstrate each of these, in turn.

**THEOREM 10.1 - Lyapunov's Direct Method**

Given a system $\dot{\mathbf{x}} = f(\mathbf{x})$, with $f$ continuous, and for some region $\mathcal{B}$ around the origin (specifically an open subset of $\mathbf{R}^n$ containing the origin), if I can produce a scalar, continuously-differentiable function $V(\mathbf{x})$, such that

$$V(\mathbf{x}) > 0, \forall \mathbf{x} \in \mathcal{B} \neq 0 \quad V(0) = 0, \text{ and}$$
$$\dot{V}(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x}) \leq 0, \forall \mathbf{x} \in \mathcal{B} \neq 0 \quad \dot{V}(0) = 0,$$

then the origin $(\mathbf{x} = 0)$ is stable in the sense of Lyapunov (i.s.L.).

If, additionally, we have

$$\dot{V}(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x}) < 0, \forall \mathbf{x} \in \mathcal{B} \neq 0,$$

then the origin is (locally) asymptotically stable. And if we have

$$\dot{V}(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x}) \leq -\alpha V(x), \forall \mathbf{x} \in \mathcal{B} \neq 0,$$

for some $\alpha > 0$, then the origin is (locally) exponentially stable.

Note that for the sequel we will use the notation $V \succ 0$ to denote a *positive-definite function*, meaning that $V(0) = 0$ and $V(\mathbf{x}) > 0$ for all $\mathbf{x} \neq 0$ (and also $V \succeq 0$ for positive semi-definite, $V \prec 0$ for negative-definite functions).

The intuition here is exactly the same as for the energy argument we made in the pendulum example: since $\dot{V}(x)$ is always zero or negative, the value of $V(x)$ will only get smaller (or stay the same) as time progresses. Inside the subset $\mathcal{B}$, for every $\epsilon$-ball, I can choose a $\delta$ such that $|x(0)|^2 < \delta \Rightarrow |x(t)|^2 < \epsilon, \forall t$ by choosing $\delta$ sufficiently small so that the sublevel-set of $V(x)$ for the largest value that $V(x)$ takes in the $\delta$ ball is completely contained in the $\epsilon$ ball. Since the value of $V$ can only get smaller (or stay constant) in time, this gives stability i.s.L.. If $\dot{V}$ is strictly negative away from the origin, then it must eventually get to the origin (asymptotic stability). The exponential condition is implied by the fact that $\forall t > 0, V(\mathbf{x}(t)) \leq V(\mathbf{x}(0))e^{-\alpha t}$.

Notice that the system analyzed above, $\dot{\mathbf{x}} = f(\mathbf{x})$, did not have any control inputs. Therefore, Lyapunov analysis is used to study either the passive dynamics of a system or the dynamics of a closed-loop system (system + control in feedback). We will see generalizations of the Lyapunov functions to input-output systems later in the text.

### 10.1.1 Global Stability

The notion of a fixed point being stable i.s.L. is inherently a local notion of stability (defined with $\epsilon$- and $\delta$- balls around the origin), but the notions of asymptotic and exponential stability can be applied globally. The Lyapunov theorems work for this case, too, with only minor modification.

**THEOREM 10.2 - Lyapunov analysis for global stability**

Given a system $\dot{\mathbf{x}} = f(\mathbf{x})$, with $f$ continuous, if I can produce a scalar, continuously-differentiable function $V(\mathbf{x})$, such that

$$V(\mathbf{x}) \succ 0,$$
$$\dot{V}(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x}) \prec 0, \text{ and}$$
$$V(\mathbf{x}) \to \infty \text{ whenever } ||x|| \to \infty,$$

then the origin ($\mathbf{x} = 0$) is globally asymptotically stable (G.A.S.).

If additionally we have that

$$\dot{V}(\mathbf{x}) \preceq -\alpha V(\mathbf{x}),$$

for some $\alpha > 0$, then the origin is globally exponentially stable.

The new condition, on the behavior as $||\mathbf{x}|| \to \infty$ is known as "radially unbounded", and is required to make sure that trajectories cannot diverge to infinity even as $V$ decreases; it is only required for global stability analysis.

### 10.1.2 LaSalle's Invariance Principle

Perhaps you noticed the disconnect between the statement above and the argument that we made for the stability of the pendulum. In the pendulum example, using the mechanical energy resulted in a Lyapunov function that was only negative semi-definite, but we eventually argued that the fixed points were asymptotically stable. That took a little extra work, involving an argument about the fact that the fixed points were the only place that the system could stay with $\dot{E} = 0$; every other state with $\dot{E} = 0$ was only transient. We can formalize this idea for the more general Lyapunov function statements--it is known as LaSalle's Theorem.

**THEOREM 10.3 - LaSalle's Theorem**

Given a system $\dot{\mathbf{x}} = f(\mathbf{x})$ with $f$ continuous. If we can produce a scalar function $V(\mathbf{x})$ with continuous derivatives for which we have

$$V(\mathbf{x}) \succ 0, \quad \dot{V}(\mathbf{x}) \preceq 0,$$

and $V(\mathbf{x}) \to \infty$ as $||\mathbf{x}|| \to \infty$, then $\mathbf{x}$ will converge to the largest *invariant set* where $\dot{V}(\mathbf{x}) = 0$.

To be clear, an *invariant set*, $\mathcal{G}$, of the dynamical system is a set for which $\mathbf{x}(0) \in \mathcal{G} \Rightarrow \forall t > 0, \mathbf{x}(t) \in \mathcal{G}$. In other words, once you enter the set you never leave. The "largest invariant set" need not be connected; in fact for the pendulum example each fixed point is an invariant set, so the largest invariant set is the *union* of all the fixed points of the system. There are also variants of LaSalle's Theorem which work over a region.

Finding a Lyapunov function which $\dot{V} \prec 0$ is more difficult than finding one that has $\dot{V} \preceq 0$. LaSalle's theorem gives us the ability to make a statement about *asymptotic* stability even in this case. In the pendulum example, every state with $\dot{\theta} = 0$ had $\dot{E} = 0$, but only the fixed points are in the largest invariant set.

### 10.1.3 Relationship to the Hamilton-Jacobi-Bellman equations

At this point, you might be wondering if there is any relationship between Lyapunov functions and the cost-to-go functions that we discussed in the context of dynamic programming. After all, the cost-to-go functions also captured a great deal about the long-term dynamics of the system in a scalar function. We can see the connection if we re-examine the HJB equation

$$0 = \min_{\mathbf{u}} \left[ g(\mathbf{x}, \mathbf{u}) + \frac{\partial J^*}{\partial \mathbf{x}} f(\mathbf{x}, \mathbf{u}). \right]$$

Let's imagine that we can solve for the optimizing $\mathbf{u}^*(\mathbf{x})$, then we are left with $0 = g(\mathbf{x}, \mathbf{u}^*) + \frac{\partial J^*}{\partial \mathbf{x}} f(\mathbf{x}, \mathbf{u}^*)$ or simply

$$\dot{J}^*(\mathbf{x}) = -g(\mathbf{x}, \mathbf{u}^*) \qquad \text{vs} \qquad \dot{V}(\mathbf{x}) \preceq 0.$$

In other words, in optimal control we must find a cost-to-go function which matches this gradient for every $\mathbf{x}$; that's very difficult and involves solving a potentially high-dimensional partial differential equation. By contrast, Lyapunov analysis is asking for much less - any function which is going downhill (at any rate) for all states. This can be much easier, for theoretical work, but also for our numerical algorithms. Also note that if we do manage to find the optimal cost-to-go, $J^*(\mathbf{x})$, then it can also serve as a Lyapunov function so long as $g(\mathbf{x}, \mathbf{u}^*(\mathbf{x})) \succeq 0$.

## 10.2 Lyapunov analysis with convex optimization

One of the primary limitations in Lyapunov analysis as I have presented it so far is that it is potentially very difficult to come up with suitable Lyapunov function candidates for interesting systems, especially for underactuated systems. ("Underactuated" is almost synonymous with "interesting" in my vocabulary.) Even if somebody were to give me a Lyapunov candidate for a general nonlinear system, the Lyapunov conditions can be difficult to check -- for instance, how would I check that $\dot{V}$ is strictly negative for all $\mathbf{x}$ except the origin if $\dot{V}$ is some arbitrarily complicated nonlinear function over a vector $\mathbf{x}$?

In this section, we'll look at some computational approaches to verifying the Lyapunov conditions, and even to searching for the description of the Lyapunov functions themselves.

If you're imagining numerical algorithms to check the Lyapunov conditions for complicated Lyapunov functions and complicated dynamics, the first thought is probably that we can evaluate $V$ and $\dot{V}$ at a large number of sample points and check whether $V$ is positive and $\dot{V}$ is negative. This does work, and could potentially be combined with some smoothness or regularity assumptions to generalize beyond the sample points. But in many cases we will be able to do better -- providing optimization algorithms that will rigorously check these conditions *for all* $\mathbf{x}$ without dense sampling; these will also give us additional leverage in formulating the search for Lyapunov functions.

### 10.2.1 Lyapunov analysis for linear systems

Let's take a moment to see how things play out for linear systems.

**Theorem 10.4 - Lyapunov analysis for stable linear systems**

Imagine you have a linear system, $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$, and can find a Lyapunov function

$$V(\mathbf{x}) = \mathbf{x}^T\mathbf{P}\mathbf{x}, \quad \mathbf{P} = \mathbf{P}^\mathbf{T} \succ 0,$$

which also satisfies

$$\dot{V}(\mathbf{x}) = \mathbf{x}^T\mathbf{P}\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{A}^T\mathbf{P}\mathbf{x} \prec 0.$$

Then the origin is globally asymptotically stable.

Note that the radially-unbounded condition is satisfied by $\mathbf{P} \succ 0$, and that the derivative condition is equivalent to the matrix condition

$$\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} \prec 0.$$

For stable linear systems the existence of a quadratic Lyapunov function is actually a necessary (as well as sufficient) condition. Furthermore, a Lyapunov function can always be found by finding the positive-definite solution to the matrix Lyapunov equation

$$\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} = -\mathbf{Q}, \tag{1}$$

for any $\mathbf{Q} = \mathbf{Q}^T \succ 0$.

This is a very powerful result - for nonlinear systems it will be potentially difficult to find a Lyapunov function, but for linear systems it is straight-forward. In fact, this result is often used to

propose candidates for non-linear systems, e.g., by linearizing the equations and solving a local linear Lyapunov function which should be valid in the vicinity of a fixed point.

### 10.2.2 Lyapunov analysis as a semi-definite program (SDP)

Lyapunov analysis for linear systems has an extremely important connection to convex optimization. In particular, we could have also formulated the Lyapunov conditions for linear systems above using *semi-definite programming* (SDP). Semidefinite programming is a subset of convex optimization -- an extremely important class of problems for which we can produce efficient algorithms that are guaranteed find the global optima solution (up to a numerical tolerance and barring any numerical difficulties).

If you don't know much about convex optimization or want a quick refresher, please take a few minutes to read the optimization preliminaries in the appendix. The main requirement for this section is to appreciate that it is possible to formulate efficient optimization problems where the constraints include specifying that one or more matrices are positive semi-definite (PSD). These matrices must be formed from a linear combination of the decision variables. For a trivial example, the optimization

$$\min_a a, \quad \text{subject to} \begin{bmatrix} a & 0 \\ 0 & 1 \end{bmatrix} \succeq 0,$$

returns $a = 0$ (up to numerical tolerances).

The value in this is immediate for linear systems. For example, we can formulate the search for a Lyapunov function for the linear system $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$ by using the parameters $\mathbf{p}$ to populate a symmetric matrix $\mathbf{P}$ and then write the SDP:

$$\underset{\mathbf{p}}{\text{find}} \quad \text{subject to} \quad \mathbf{P} \succeq 0, \quad \mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} \preceq 0.$$

Note that you would probably never use that particular formulation, since there specialized algorithms for solving the simple Lyapunov equation which are more efficient and more numerically stable. But the SDP formulation does provide something new -- we can now easily formulate the search for a *"common Lyapunov function"* for uncertain linear systems.

**EXAMPLE 10.2 (Common Lyapunov analysis for linear systems)**

Suppose you have a system governed by the equations $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$, where the matrix $\mathbf{A}$ is unknown, but its uncertain elements can be bounded. There are a number of ways to write down this uncertainty set; let us choose to write this by describing $\mathbf{A}$ as the convex combination of a number of known matrices,

$$\mathbf{A} = \sum_i \beta_i \mathbf{A}_i, \quad \sum_i \beta_i = 1, \quad \forall i, \beta_i > 0.$$

This is just one way to specify the uncertainty; geometrically it is describing a polygon of uncertain parameters (in the space of elements of $\mathbf{A}$ with each $\mathbf{A}_i$ as one of the vertices in the polygon.

Now we can formulate the search for a common Lyapunov function using

$$\underset{\mathbf{P}}{\text{find}} \quad \text{subject to} \quad \mathbf{P} \succeq 0, \quad \forall_i, \mathbf{P}\mathbf{A}_i + \mathbf{A}_i^T\mathbf{P} \preceq 0.$$

The solver will then return a matrix $\mathbf{P}$ which satisfies all of the constraints, or return saying "problem is infeasible". It can easily be verified that if $\mathbf{P}$ satisfies the Lyapunov condition at all of the vertices, then it satisfies the condition for every $\mathbf{A}$ in the set:

$$\mathbf{P}(\sum_i \beta_i \mathbf{A}_i) + (\sum_i \beta_i \mathbf{A}_i)^T\mathbf{P} = \sum_i \beta_i(\mathbf{P}\mathbf{A}_i + \mathbf{A}_i^T\mathbf{P}) \preceq 0,$$

since $\forall i, \beta_i > 0$. Note that, unlike the simple Lyapunov equation for a known linear system, this condition being satisfied is a sufficient but not a necessary condition -- it is possible that the set of uncertain matrices $\mathbf{A}$ is robustly stable, but that this stability cannot be demonstrated with a common quadratic Lyapunov function.

You can try this example for yourself in DRAKE.

```
python lyapunov/linear_systems_common_lyapunov.py
```

Cola

As always, make sure that you open up the code and take a look.

This example is very important because it establishes a connection between Lyapunov functions and (convex) optimization. But so far we've only demonstrated this connection for linear systems where the PSD matrices provide a magical recipe for establishing the positivity of the (quadratic) functions for all $\mathbf{x}$. Is there any hope of extending this type of analysis to more general nonlinear systems? Surprisingly, it turns out that there is.

### 10.2.3 Sums-of-squares optimization

It turns out that in the same way that we can use SDP to search over the positive quadratic equations, we can generalize this to search over the positive polynomial equations. To be clear, for quadratic equations we have

$$\mathbf{P} \succeq 0 \quad \Rightarrow \quad \mathbf{x}^T\mathbf{P}\mathbf{x} \geq 0, \quad \forall \mathbf{x}.$$

It turns out that we can generalize this to

$$\mathbf{P} \succeq 0 \quad \Rightarrow \quad \mathbf{m}^T(\mathbf{x})\mathbf{P}\mathbf{m}(\mathbf{x}) \geq 0, \quad \forall \mathbf{x},$$

where $\mathbf{m}(\mathbf{x})$ is a vector of polynomial equations, typically chosen as a vector of *monomials* (polynomials with only one term). The set of positive polynomials parameterized in this way is exactly the set of polynomials that can be written as a *sum of squares*[62]. While it is known that not all positive polynomials can be written in this form, much is known about the gap (and papers have been written about trying to find polynomials which are uniformly positive but not sums of

squares). For our purposes this gap is very small; we should remember that it exists but not worry about it too much for now.

Even better, there is quite a bit that is known about how to choose the terms in $\mathbf{m}(\mathbf{x})$. For example, if you give me the polynomial

$$p(x) = 2 - 4x + 5x^2$$

and ask if it is positive for all real $x$, I can convince you that it is by producing the sums-of-squares factorization

$$p(x) = 1 + (1 - 2x)^2 + x^2,$$

and I know that I can formulate the problem without needing any monomials of degree greater than 1 (the square-root of the degree of $p$) in my monomial vector. In practice, what this means for us is that people have authored optimization front-ends which take a high-level specification with constraints on the positivity of polynomials and they automatically generate the SDP problem for you, without having to think about what terms should appear in $\mathbf{m}(\mathbf{x})$ (c.f. [63]). This class of optimization problems is called Sums-of-Squares (SOS) optimization.

As it happens, the particular choice of $\mathbf{m}(\mathbf{x})$ can have a huge impact on the numerics of the resulting semidefinite program (and on your ability to solve it with commercial solvers). DRAKE implements some particularly novel/advanced algorithms to make this work well [64].

We will write optimizations using sums-of-squares constraints as

$$p(\mathbf{x}) \text{ is SOS}$$

as shorthand for the constraint that $p(\mathbf{x}) \geq 0$ for all $\mathbf{x}$, as demonstrated by finding a sums-of-squares decomposition.

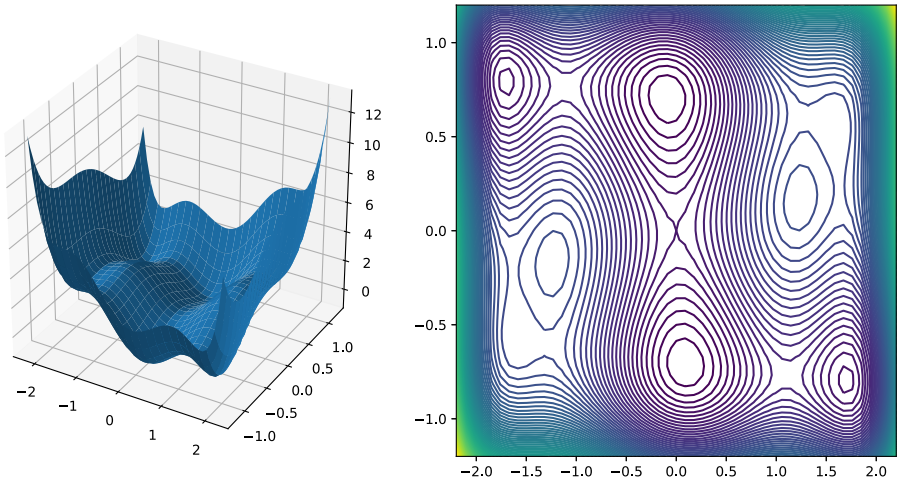
This is surprisingly powerful. It allows us to use convex optimization to solve what appear to be very non-convex optimization problems.

**EXAMPLE 10.3 (Global minimization via SOS)**

Consider the following famous non-linear function of two variables (called the "six-hump-camel":

$$p(x) = 4x^2 + xy - 4y^2 - 2.1x^4 + 4y^4 + \frac{1}{3}x^6.$$

This function has six local minima, two of them being global minima [65].




By formulating a simple sums-of-squares optimization, we can actually find the minimum value of this function (technically, it is only a lower bound, but in this case and many cases, it is

surprisingly tight) by writing:

$$\max_{\lambda} \quad \lambda$$
$$\text{s.t. } p(x) - \lambda \text{ is sos.}$$

Go ahead and play with the code (most of the lines are only for plotting; the actual optimization problem is nice and simple to formulate).

```
python simple/six_hump_camel.py
```

Cola

Note that this finds the minimum value, but does not actually produce the $\mathbf{x}$ value which mimimizes it. This is possible [65], but it requires examining the dual of the sums-of-squares solutions (which we don't need for the goals of this chapter).

### 10.2.4 Lyapunov analysis for polynomial systems

Now we can see that it may be possible to generalize the optimization approach using SDP to search for Lyapunov functions for linear systems to searching for Lyapunov functions for at least the polynomial systems: $\dot{\mathbf{x}} = f(\mathbf{x})$, where $f$ is a vector-valued polynomial function. If we parameterize a *fixed-degree* Lyapunov candidate as a polynomial with unknown coefficients, e.g.,

$$V_\alpha(\mathbf{x}) = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_1 x_2 + \alpha_4 x_1^2 + \ldots,$$

then the time-derivative of $V$ is also a polynomial, and I can formulate the optimization:

$$\underset{\alpha}{\text{find}}, \quad \text{subject to} \quad V_\alpha(\mathbf{x}) \text{ is SOS}$$
$$-\dot{V}_\alpha(\mathbf{x}) = -\frac{\partial V_\alpha}{\partial \mathbf{x}} f(\mathbf{x}) \text{ is SOS}.$$

Because this is a convex optimization, the solver will return a solution if one exists.

**Example 10.4 (Verifying a Lyapunov candidate via SOS)**

This example is example 7.2 from [62]. Consider the nonlinear system:

$$\dot{x}_0 = -x_0 - 2x_1^2$$
$$\dot{x}_1 = -x_1 - x_0 x_1 - 2x_1^3,$$

and the *fixed* Lyapunov function $V(x) = x_0^2 + 2x_1^2$, test if $\dot{V}(x)$ is negative definite.

The numerical solution can be written in a few lines of code, and is a convex optimization.

```
python lyapunov/verify_global_polynomial.py
```

Cola

**Example 10.5 (Searching for a Lyapunov function via SOS)**

Verifying a candidate Lyapunov function is all well and good, but the real excitement starts when we use optimization to *find* the Lyapunov function. In the following code, we parameterize $V(x)$ as the polynomial containing all monomials up to degree 2, with the coefficients as decision variables:

$$V(x) = c_0 + c_1 x_0 + c_2 x_1 + c_3 x_0^2 + c_4 x_0 x_1 + c_5 x_1^2.$$

We will set the scaling (arbitrarily) to avoid numerical issues by setting $V(0) = 0$, $V([1, 0]) = 1$. Then we write:

$$\begin{aligned} \underset{\mathbf{c}}{\text{find}} \quad & \text{subject to } V \text{ is sos}, \\ & -\dot{V} \text{ is sos}. \end{aligned}$$

```
python lyapunov/search_global_polynomial.py
```

Cola

Up to numerical convergence tolerance, it discovers the same coefficients that we chose above (zeroing out the unnecessary terms).

It is important to remember that there are a handful of gaps which make the existence of this solution a sufficient condition (for proving that every sub-level set of $V$ is an invariant set of $f$) instead of a necessary one. First, there is no guarantee that a stable polynomial system can be verified using a polynomial Lyapunov function (of any degree, and in fact there are known counter-examples [66]) and here we are only searching over a fixed-degree polynomial. Second, even if a polynomial Lyapunov function does exist, there is a gap between the SOS polynomials and the positive polynomials.

Despite these caveats, I have found this formulation to be surprisingly effective in practice. Intuitively, I think that this is because there is relatively a lot of flexibility in the Lyapunov conditions -- if you can find one function which is a Lyapunov function for the system, then there are also many "nearby" functions which will satisfy the same constraints.

## 10.3 Lyapunov functions for estimating regions of attraction

There is another very important connection between Lyapunov functions and the concept of an invariant set: *any sub-level set of a Lyapunov function is also an invariant set*. This gives us the ability to use sub-level sets of a Lyapunov function as approximations of the region of attraction for nonlinear systems.

**Theorem 10.5 - Lyapunov invariant set and region of attraction theorem**

Given a system $\dot{\mathbf{x}} = f(\mathbf{x})$ with $f$ continuous, if we can find a scalar function $V(\mathbf{x}) \succ 0$ and a sub-level set

$$\mathcal{G} : \{\mathbf{x} | V(\mathbf{x}) < \rho\}$$

on which

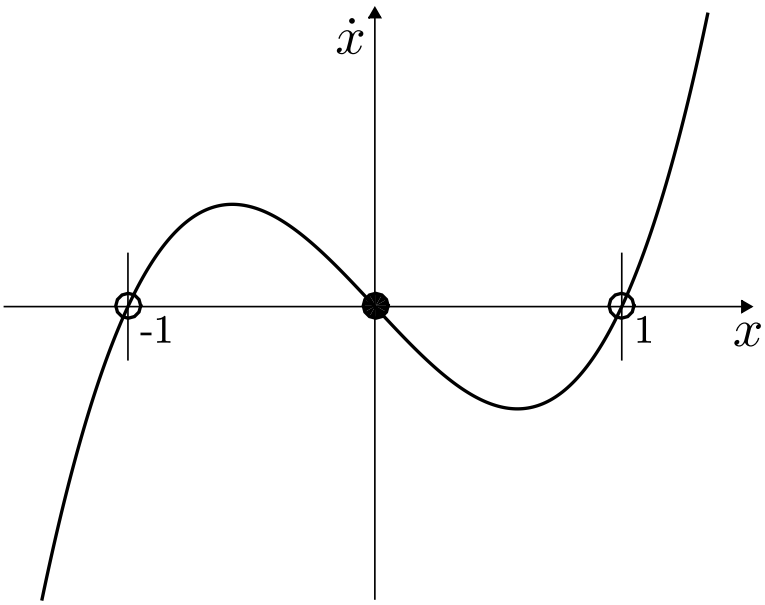
$$\forall \mathbf{x} \in \mathcal{G}, \dot{V}(\mathbf{x}) \preceq 0,$$

then $\mathcal{G}$ is an invariant set. By LaSalle, $\mathbf{x}$ will converge to the largest invariant subset of $\mathcal{G}$ on which $\dot{V} = 0$.

Furthermore, if $\dot{V}(\mathbf{x}) \prec 0$ in $\mathcal{G}$, then the origin is locally asymptotically stable and the set $\mathcal{G}$ is inside the region of attraction of this fixed point. Alternatively, if $\dot{V}(\mathbf{x}) \preceq 0$ in $\mathcal{G}$ and $\mathbf{x} = 0$ is the only invariant subset of $\mathcal{G}$ where $\dot{V} = 0$, then the origin is asymptotically stable and the set $\mathcal{G}$ is inside the region of attraction of this fixed point.

**Example 10.6 (Region of attraction for a one-dimensional system)**

Consider the first-order, one-dimensional system $\dot{x} = -x + x^3$. We can quickly understand this system using our tools for graphical analysis.




In the vicinity of the origin, $\dot{x}$ looks like $-x$, and as we move away it looks increasingly like $x^3$. There is a stable fixed point at the origin and unstable fixed points at $\pm 1$. In fact, we can deduce visually that the region of attraction to the stable fixed point at the origin is $\mathbf{x} \in (-1, 1)$. Let's see if we can demonstrate this with a Lyapunov argument.

First, let us linearize the dynamics about the origin and use the Lyapunov equation for linear systems to find a candidate $V(\mathbf{x})$. Since the linearization is $\dot{x} = -x$, if we take $\mathbf{Q} = 1$, then we find $\mathbf{P} = \frac{1}{2}$ is the positive definite solution to the algebraic Lyapunov equation (1). Proceeding with

$$V(\mathbf{x}) = \frac{1}{2}x^2,$$

we have

$$\dot{V} = x(-x + x^3) = -x^2 + x^4.$$

This function is zero at the origin, negative for $|x| < 1$, and positive for $|x| > 1$. Therefore we can conclude that the sub-level set $V < \frac{1}{2}$ is invariant and the set $x \in (-1, 1)$ is inside the region of attraction of the nonlinear system. In fact, this estimate is tight.

### 10.3.1 Robustness analysis using "common Lyapunov functions"

While we will defer most discussions on robustness analysis until later in the notes, there is a simple and powerful idea that can be presented now: the idea of a *common Lyapunov function*. Imagine that you have a model of a dynamical system but that you are uncertain about some of the parameters. For example, you have a model of a quadrotor, and are fairly confident about the mass and lengths (both of which are easy to measure), but are not confident about the moment of inertia. One approach to robustness analysis is to define a bounded uncertainty, which could take the form of

$$\dot{\mathbf{x}} = f_\alpha(\mathbf{x}), \quad \alpha_{min} \leq \alpha \leq \alpha_{max},$$

with $\alpha$ a vector of uncertain parameters in your model. Richer specifications of the uncertainty bounds are also possible, but this will serve for our examples.

In standard Lyapunov analysis, we are searching for a function that goes downhill for all $\mathbf{x}$ to make statements about the long-term dynamics of the system. In common Lyapunov analysis, we can make many similar statements about the long-term dynamics of an uncertain system if we can find a single Lyapunov function that goes downhill *for all possible values of* $\alpha$. If we can find such a

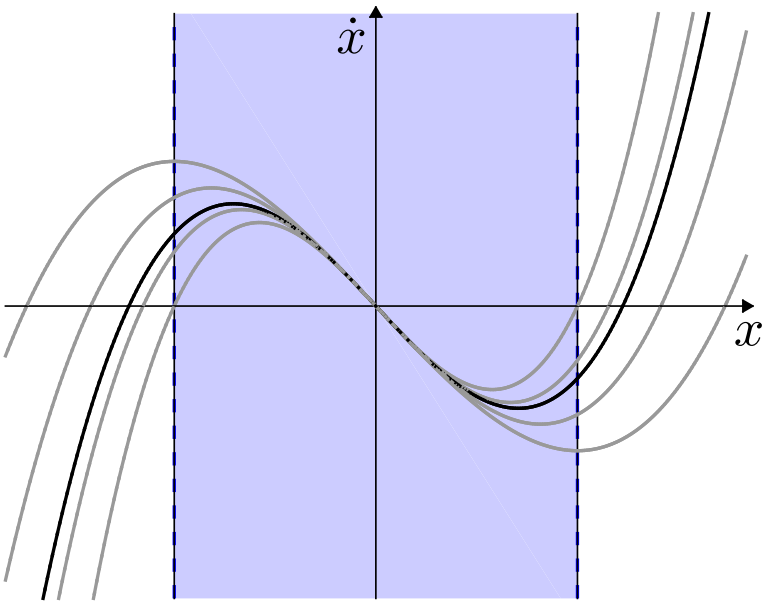
function, then we can use it to make statements with all of the variations we've discussed (local, regional, or global; in the sense of Lyapunov, asymptotic, or exponential).

**EXAMPLE 10.7 (A one-dimensional system with gain uncertainty)**

Let's consider the same one-dimensional example used above, but add an uncertain parameter into the dynamics. In particular, consider the system:

$$\dot{x} = -x + \alpha x^3, \quad \frac{3}{4} < \alpha < \frac{3}{2}.$$

Plotting the graph of the one-dimensional dynamics for a few values of $\alpha$, we can see that the fixed point at the origin is still stable, but the *robust region of attraction* to this fixed point (shaded in blue below) is smaller than the region of attraction for the system with $\alpha = 1$.




Taking the same Lyapunov candidate as above, $V = \frac{1}{2}x^2$, we have

$$\dot{V} = -x^2 + \alpha x^4.$$

This function is zero at the origin, and negative for all $\alpha$ whenever $x^2 > \alpha x^4$, or

$$|x| < \frac{1}{\sqrt{\alpha_{max}}} = \sqrt{\frac{2}{3}}.$$

Therefore, we can conclude that $|x| < \sqrt{\frac{2}{3}}$ is inside the robust region of attraction of the uncertain system.

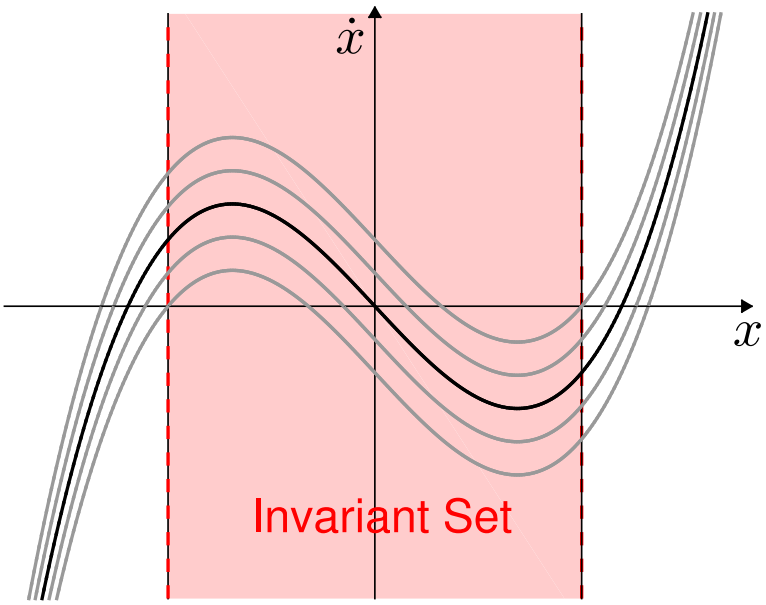
Not all forms of uncertainty are as simple to deal with as the gain uncertainty in that example. For many forms of uncertainty, we might not even know the location of the fixed points of the uncertain system. In this case, we can often still use common Lyapunov functions to give some guarantees about the system, such as guarantees of *robust set invariance*. For instance, if you have uncertain parameters on a quadrotor model, you might be ok with the quadrotor stabilizing to a pitch of 0.01 radians, but you would like to guarantee that it definitely does not flip over and crash into the ground.

**EXAMPLE 10.8 (A one-dimensional system with additive uncertainty)**

Now consider the system:

$$\dot{x} = -x + x^3 + \alpha, \quad -\frac{1}{4} < \alpha < \frac{1}{4}.$$

Plotting the graph of the one-dimensional dynamics for a few values of $\alpha$, this time we can see that the fixed point is not necessarily at the origin; the location of the fixed point moves depending on the value of $\alpha$. But we should be able to guarantee that the uncertain system will stay near the origin if it starts near the origin, using an invariant set argument.




Taking the same Lyapunov candidate as above, $V = \frac{1}{2}x^2$, we have

$$\dot{V} = -x^2 + x^4 + \alpha x.$$

This Lyapunov function allows us to easily verify, for instance, that $V \leq \frac{1}{3}$ is a *robust invariant set*, because whenever $V = \frac{1}{3}$, we have

$$\forall \alpha \in [\alpha_{min}, \alpha_{max}], \quad \dot{V}(x, \alpha) < 0.$$

Therefore $V$ can never start at less than one-third and cross over to greater than one-third. To see this, see that

$$V = \frac{1}{3} \quad \Rightarrow \quad x = \pm\sqrt{\frac{2}{3}} \quad \Rightarrow \quad \dot{V} = -\frac{2}{9} \pm \alpha\sqrt{\frac{2}{3}} < 0, \forall \alpha \in \left[-\frac{1}{4}, \frac{1}{4}\right].$$

Note that not all sub-level sets of this invariant set are invariant. For instance $V < \frac{1}{32}$ does not satisfy this condition, and by visual inspection we can see that it is in fact not robustly invariant.

### 10.3.2 Region of attraction estimation for polynomial systems

Now we have arrived at the tool that I believe can be a work-horse for many serious robotics applications. Most of our robots are not actually globally stable (that's not because they are robots -- if you push me hard enough, I will fall down, too), which means that understanding the regions where a particular controller can be guaranteed to work can be of critical importance.

Sums-of-squares optimization effectively gives us an oracle which we can ask: is this polynomial positive for all $\mathbf{x}$? To use this for regional analysis, we have to figure out how to modify our questions to the oracle so that the oracle will say "yes" or "no" when we ask if a function is positive over a certain region which is a subset of $\Re^n$. That trick is called the S-procedure. It is closely related to the Lagrange multipliers from constrained optimization, and has deep connections to "Positivstellensatz" from algebraic geometry.

### The S-procedure

Consider a scalar polynomial, $p(\mathbf{x})$, and a semi-algebraic set $g(\mathbf{x}) \preceq 0$, where $g$ is a vector of polynomials. If I can find a polynomial "multiplier", $\lambda(\mathbf{x})$, such that

$$p(\mathbf{x}) + \lambda^T(\mathbf{x})g(\mathbf{x}) \text{ is SOS}, \quad \text{and} \quad \lambda(\mathbf{x}) \text{ is SOS},$$

then this is sufficient to demonstrate that

$$p(\mathbf{x}) \geq 0 \quad \forall \mathbf{x} \in \{g(\mathbf{x}) \leq 0\}.$$

To convince yourself, observe that when $g(\mathbf{x}) \leq 0$, it is only harder to be positive, but when $g(\mathbf{x}) > 0$, it is possible for the combined function to be SOS even if $p(\mathbf{x})$ is negative.

We can also handle equality constraints with only a minor modification -- we no longer require the multiplier to be positive. If I can find a polynomial "multiplier", $\lambda(\mathbf{x})$, such that

$$p(\mathbf{x}) + \lambda^T(\mathbf{x})g(\mathbf{x}) \text{ is SOS}$$

then this is sufficient to demonstrate that

$$p(\mathbf{x}) \geq 0 \quad \forall \mathbf{x} \in \{g(\mathbf{x}) = 0\}.$$

Here the intuition is that $\lambda(x)$ can add arbitrary positive terms to help me be SOS, but those terms contribute nothing precisely when $g(x) = 0$.

### Basic region of attraction formulation

**Example 10.9 (Region of attraction for the one-dimensional cubic system)**

Let's return to our example from above:

$$\dot{x} = -x + x^3$$

and try to use SOS optimization to demonstrate that the region of attraction of the fixed point at the origin is $x \in (-1, 1)$, using the Lyapunov candidate $V = x^2$.

First, define the multiplier polynomial,

$$\lambda(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + c_4 x^4.$$

Then define the optimization

$$\begin{aligned} \underset{\mathbf{c}}{\text{find}} & \\ \text{subject to} \quad & -\dot{V}(x) - \lambda(x)(1 - V(x)) \text{ is SOS} \\ & \lambda(x) \text{ is SOS} \end{aligned}$$

You can try this example for yourself in Drake.

```
python lyapunov/cubic_polynomial.py
```

Cola

While the example above only verifies that the one-sub-level set of the pre-specified Lyapunov candidate is negative (certifying the ROA that we already understood), we can generalize the optimization to allow us to search for the largest sub-level set (with the objective using a convex approximation for volume). We can even search for the coefficients of the Lyapunov function using an iteration of convex optimizations. There are a number of variations and nuances in the various formulations, and some basic rescaling tricks that can help make the numerics of the problem better for the solvers.

### Seeding the optimization with linear analysis

LQR gives the cost-to-go which can be used as the Lyapunov candidate. Otherwise, use a Lyapunov equation. You may not even need to search for a better Lyapunov function, but rather just ask the question: what is the largest region of attraction that can be demonstrated for the nonlinear system using the Lyapunov function from linear analysis?

### 10.3.3 Rigid-body dynamics are polynomial

Coming soon... For a nice explanation of how rigid-body kinematics are polynomial, see [67]. Looking at the Lagrangian, you can see that if the kinematics are polynomial, then the dynamics will be, too.

**EXAMPLE 10.10 (Global stability of the simple pendulum via SOS)**

We opened this chapter using our intuition about energy to discuss stability on the simple pendulum. Now we'll replace that intuition with convex optimization (because it will also work for more difficult systems where our intuition fails).

But wait! The equations of motion of the simple pendulum are not polynomial, are they (they have a $\sin(\theta)$ in them)? Actually nearly all rigid-body systems can be written as polynomials given a change of coordinates. Indeed, the rigid-body-assumption is simply an assumption that points on the same body stay a constant distance apart, and Euclidean distance is certainly a polynomial. For a more thorough discussion see, for instance, [67] and [68].

For the purposes of this example, let's change coordinates from $[\theta, \dot{\theta}]^T$ to $\mathbf{x} = [s, c, \dot{\theta}]^t$, where $s \equiv \sin\theta$ and $c \equiv \cos\theta$. Then we can write the pendulum dynamics as

$$\dot{\mathbf{x}} = \begin{bmatrix} c\dot{\theta} \\ -s\dot{\theta} \\ -\frac{1}{ml^2}\left(b\dot{\theta} + mgls\right) \end{bmatrix}.$$

Now let's parameterize a Lyapunov candidate $V(s, c, \dot{\theta})$ as the polynomial with unknown coefficients which contains all monomials up to degree 2:

$$V = \alpha_0 + \alpha_1 s + \alpha_2 c + \ldots \alpha_9 s^2 + \alpha_{10} sc + \alpha_{11} s\dot{\theta}.$$

Now we'll formulate the feasibility problem:

$$\underset{\alpha}{\text{find}} \quad \text{subject to} \quad V \text{ is SOS}, \quad -\dot{V} \text{ is SOS}.$$

In fact, this is asking too much -- really $\dot{V}$ only needs to be negative when $s^2 + c^2 = 1$. We can accomplish this with the S-procedure, and instead write

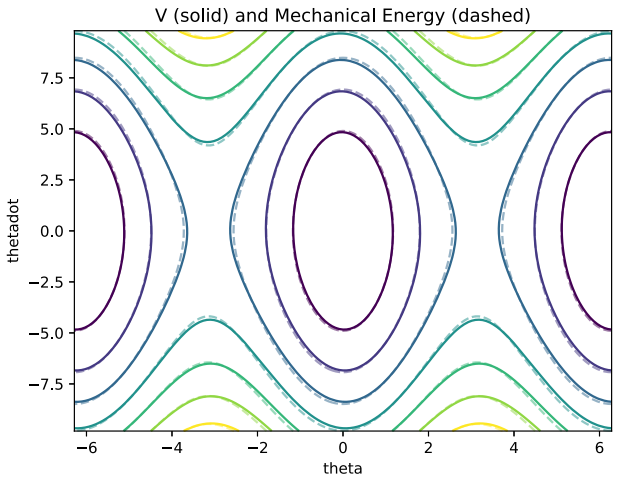
$$\underset{\alpha,\lambda}{\text{find}} \quad \text{subject to} \quad V \text{ is SOS}, \quad -\dot{V} - \lambda(\mathbf{x})(s^2 + c^2 - 1) \text{ is SOS}.$$

(Recall that $\lambda(\mathbf{x})$ is another polynomial with free coefficients which the optimization can use to make terms arbitrarily more positive when $s^2 + c^2 \neq 1$.) Finally, for style points, in the code example in DRAKE we ask for exponential stability:

```
python pendulum/global_sums_of_squares.py
```

Cola

As always, make sure that you open up the code and take a look. The result is a Lyapunov function that looks familiar (visualized as a contour plot here):




Aha! Not only does the optimization finds us coefficients for the Lyapunov function which satisfy our Lyapunov conditions, but the result looks a lot like mechanical energy. In fact, the result is a little better than energy... there are some small extra terms added which prove exponential stability without having to invoke LaSalle's Theorem.

In practice, you can also Taylor approximate any smooth nonlinear function using polynomials. This can be an effective strategy in practice, because you can limit the degrees of the polynomial, and because in many cases it is possible to provide conservative bounds on the errors due to the approximation.

Previous chapter Table of contents Next chapter