6.832 Final Project: A Study of Contraction Analysis for Stability Analysis and Controller Synthesis via Sum-of-Squares Programming

Sunbochen Tang

Abstract

The objective of this project is to explore contraction analysis-based methods for nonlinear system stability analysis and controller synthesis using Sums-of-squares Programming. Main theoretical results in the literature including recent seminal work [1] are revisited with a focus on building necessary understanding of Riemannian geometry, contraction analysis, and relationships between different theorems and their conditions. We describe the algorithmic procedure to search for contraction metrics, compute geodesics, and synthesize controllers. Simulation results for various polynomial systems are also developed which verify the correctness of our implementation and identify several challenges with this approach.

I. Introduction

Contraction analysis provides an alternative perspective into stability, namely, incremental stability between two arbitrary trajectories. By studying the differential dynamics, contraction analysis reduces stability analysis of nonlinear systems to its differential counterpart, which is linear time-varying. As a result, convex conditions on the differential dynamics can be derived to certify asymptotic or exponential convergence of two arbitrary trajectories of a stable autonomous system.

More interestingly, contraction analysis also enables control design that achieves incremental stability, analogous to the control Lyapunov function approach in the Lyapunov function framework. For a class of control-affine systems, the fact that its differential dynamics is linear time-varying leads to a differential feedback controller design, which guarantees exponential convergence to a reference trajectory. Such tracking controller is of interests to many industrial applications especially in robotics where a hierarchical decision making structure containing a planning layer that searches for a reference trajectory and a control layer that tracks said reference trajectory is widely adopted. Furthermore, compared to online optimization approaches such as Model Predictive Control (MPC), control contraction metrics computation can be completed offline, thus reducing online computation to a task of finding the geodesic, which does not have dynamical constraint hence much more efficient than MPC.

Fascinated by all these benefits (and to be honest, the Riemannian geometry mathematical notion behind it), the objective of my project is to understand this framework along with an algorithmic implementation procedure, and develop numerical examples that use contraction analysis for stability analysis and tracking controller design. The project is closely related to course contents including SOS programming, nonlinear system stability, Lyapunov function, trajectory optimization. The project might be balanced between theoretical result understanding and algorithmic implementation, as at least for me the learning curve for the contraction analysis framework is a bit deep.

II. Related Work

The idea of using contraction theory for nonlinear system stability analysis has been well studied in the literature, with notable reference such as [2]. In [3], a Sum-of-squares programming-based algorithmic search for contraction metrics is proposed, which reduces searching for infinite-dimensional contraction metric function satisfying LMI conditions in [2] into a convex programming problem.

More recently, in [4], [1], the framework is extended to synthesize controller that provides exponential convergence to a target trajectory. More importantly, [4] proposes a convex condition on the contraction metrics based on a differential feedback controller design, which reduces control contraction metric search to a convex programming problem. In [1], the results are extended to open-loop controller design, and sampled-feedback controller design. In addition, the past few years has seen a variety of approaches to contraction analysis using machine learning. The key idea is instead of using polynomial basis functions, one can use neural-contraction metrics trained using offline data. Notable references include [5], [6], [7].

III. Contraction Analysis Theoretical Results

In this section, we revisit theoretical results in contraction analysis literature. The following summary is my personal "roadmap" of known results, which omits the proof of theorems (reference provided for readers), and focuses on the connection between known results and developing personal insights. Texts in blue color refer to discussion or connections made outside contents in the literature, and texts in black color is either paraphrase or quotes of re-organized known results in literature.

A. Preliminaries: Riemannian geometry [1]

Let $x_1(t)$ and $x_2(t)$ be two arbitrary smooth trajectories of a dynamical system with state vector $x \in \mathbb{R}^n$. At time instant t, $\Gamma(x_1,x_2,t)$ denotes the set of all smooth paths connecting $x_1(t)$ and $x_2(t)$, where each path $c(s,t) \in \Gamma(x_1,x_2,t)$ is parameterized by $s \in [0,1]$, and satisfies $c(0,t) = x_1(t)$, $c(1,t) = x_2(t)$.

A Riemannian metric is a smoothly varying inner product on the tangent space of the state manifold $\mathscr{X} = \mathbb{R}^n$. Such a metric is important because it defines the notion of length, angle, energy of a differential element of the path, denoted as $c_s(s) := \frac{\partial c(s)}{\partial s}$. Let $M(x,t)$ be a Riemannian metric, we use the following notations,

$$< c_s, c_s >_{c,t} = c_s^T M(x,t) c_s, \quad x = c(s)$$
$$\|c_s\|_{c,t} = \sqrt{< c_s, c_s >_{c,t}} = \sqrt{c_s^T M(x,t) c_s}$$

For a smooth curve $c : [0,1] \to \mathbb{R}^n \in \Gamma$ the Riemannian length $L(c,t)$ and energy functions $E(c,t)$ can be defined as:

$$L(c,t) = \int_0^1 \|c_s\|_{c,t} ds, \quad E(c,t) = \int_0^1 \|c_s\|_{c,t}^2 ds \tag{1}$$

The Hopf-Rinow Theorem implies a smooth regular minimum-length curve (a.k.a. "geodesic"), $\gamma(s,t) \in \Gamma(x_1, x_2, t)$, exists at any given t and it satisfies

$$\gamma_s = \frac{\partial \gamma(s)}{\partial s} \neq 0, \quad L(\gamma,t) = \sqrt{E(\gamma,t)} = \inf_{c \in \Gamma} L(c,t)$$

Note that the geodesic has constant speed (Proof of Lemma 1 in [1]), i.e., $\|\gamma_s\|_{\gamma,t}$ is constant. This result implies the following inequalities:

$$E(\gamma,t) = L(\gamma,t)^2 \leq L(c,t)^2 \leq E(c,t), \quad \forall c \in \Gamma$$

These inequalities are very important for geodesic computation (discussed in details in Section IV.B. They are not trivial because $M(x,t)$ can be both spatially and time-varying. Here we provide a short proof of the above inequalities: Since $||\gamma_s||_{\gamma,t}$ is constant, denote the constant as D_0, then we have

$$E(\gamma,t) = \int_0^1 D_0^2 ds = D_0^2 = \left(\int_0^1 D_0 ds \right)^2 = L(\gamma,t)^2$$

By Cauchy-Schwartz inequality and the fact that $0 \leq L(\gamma,t) = \inf_{c \in \Gamma} L(c,t)$,

$$\forall c \in \Gamma, \quad E(\gamma,t) = L(\gamma,t)^2 \leq L(c,t)^2 = \left(\int_0^1 \|c_s\|_{c,t} ds \right)^2$$
$$\leq \int_0^1 \|c_s\|_{c,t}^2 ds = E(c,t)$$

The above inequality states that γ must be a global minimizer of the Riemannian energy of all paths, i.e., $\gamma = argmin_c E(c,t)$.

B. Contraction Analysis for Nonlinear System Stability

Consider an autonomous nonlinear system:

$$\dot{x} = f(x,t) \tag{2}$$

where $x \in \mathbb{R}^n$ is the state vector. Before jumping into the differential dynamics and their connection to the Riemannian geometry, we discuss the motivation behind introducing a spatial and time-varying Riemannian metric $M(x,t)$.

The goal of introducing incremental stability is to reduce the convergence of two arbitrary trajectories to the problem of decreasing distance of every differential element of the path that connects these trajectories. Suppose every differential element already has a decreasing Euclidean distance, i.e., $\frac{d}{dt}\sqrt{\gamma_s^T \gamma_s} < 0$, then the Euclidean distance $\|x_1(t) - x_2(t)\|$ will also be decreasing, thus showing convergence. However, this is not generally true even for two converging trajectories. In some sense, $M(x,t)$ relaxes the convergence condition, instead of asking for decreasing Euclidean distance, we now use Riemannian distance under a metric $M(x,t)$. Once such a metric is found, the geodesic length will decrease over time, thus proving convergence.

Formally, following the notation in the last subsection, for any two arbitrary trajectories of this system $x_1(t), x_2(t)$, assume there exists a geodesic $\gamma(s,t)$ and denote the differential element of the geodesic as $\delta x := \gamma_s$.

The differential dynamics of (2) is linear time-varying,

$$\dot{\delta x} = \frac{\partial f}{\partial x} \delta x, \tag{3}$$

Proposition 1: Consider the system in (2), if a metric function $M(x,t)$ satisfies the following conditions, then any two trajectories $x_1(t), x_2(t)$ will converge asymptotically, i.e., $\lim_{t \to \infty} \|x_1(t) - x_2(t)\| = 0$. [3]

1) $M(x,t)$ must be a uniformly positive definite matrix, i.e., $\exists \varepsilon > 0$,

$$M(x,t) \succeq \varepsilon I \succ 0, \quad \forall x,t \tag{4}$$

2) The metric variation $\frac{\partial f}{\partial x}^T M + M \frac{\partial f}{\partial x} + \dot{M}$ must be a uniformly negative definite matrix, i.e., $\exists \varepsilon > 0$,

$$R(x,t) = \frac{\partial f}{\partial x}^T M + M \frac{\partial f}{\partial x} + \dot{M} \preceq -\varepsilon I \prec 0, \forall x,t \tag{5}$$

Proof: (Sketch) Suppose (4) and (5) hold, then the differential element length under the Riemannian metric $M(x,t)$ satisfies

$$\frac{d}{dt}\left(\delta x^T M \delta x\right) = \delta x^T \left(\frac{\partial f}{\partial x}^T M + M \frac{\partial f}{\partial x} + \dot{M} \right) \delta x < 0$$

Since $M(x,t)$ is uniformly positive definite, the differential element must all converge to 0, $\lim_{t \to \infty} \|x_1(t) - x_2(t)\| = 0$. ■

Remark 1: Suppose instead of (5), the following condition hold: $\exists \varepsilon, \lambda > 0$

$$R(x,t) = \frac{\partial f}{\partial x}^T M + M \frac{\partial f}{\partial x} + \dot{M} + 2\lambda M \preceq -\varepsilon I \prec 0, \forall x,t \tag{6}$$

Then, any two trajectories will converge exponentially with rate λ as $\frac{d}{dt}\left(\delta x^T M \delta x\right) \leq -2\lambda \left(\delta x^T M \delta x\right)$.

Note that the contraction metric is not unique, and its existence is a sufficient condition to trajectory convergence, not necessary. The formal stability definition will be introduced in the next subsection with control-affine systems.

C. Control Contraction Metrics - Controller Synthesis

Now we consider a control problem where contraction analysis is leveraged to synthesize a controller that tracks a reference trajectory. Consider a class of control-affine systems,

$$\dot{x} = f(x,t) + B(x,t)u \tag{7}$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ are state and control vector, respectively. Denote the i-th column of $B(x,t)$ as $b_i(x,t)$. The differential dynamics of (7) is,

$$\dot{\delta}_x = A(x,u,t)\delta_x + B(x,t)\delta_u, \quad A = \frac{\partial}{\partial x} + \sum_{i=1}^{m} \frac{\partial b_i}{\partial x} u_i \tag{8}$$

In this report, we follow the notation used in [1] for control contraction metrics because the framework is more complete in the sense of including sampled-data feedback controller and open-loop controller. However, personally the theoretical results in [4] are found to be more streamlined and easier to follow, albeit its slight simplicity of a constant control effectiveness matrix $B(t)$. In order to make this report self-contained, the main results in [1] are quoted as follows.

Definition 1: [1] A target trajectory (x^*, u^*) is a forward-complete solution of (7). The target trajectory (x^*, u^*) is said to be globally exponentially controllable (respectively, stabilizable) if one can construct an open-loop (respectively, feedback) controller such that for any initial condition $x(0) \in \mathbb{R}^n$, a unique solution $x(t)$ of (7) exists for all t and satisfies

$$\|x(t) - x^*(t)\| \leq e^{-\lambda t} R \|x(0) - x^*(0)\|$$

where rate $\lambda > 0$ and overshoot $R > 0$ are constants independent of initial conditions. If every target trajectory is globally exponentially controllable (respectively, stabilizable), then the system is said to be universally universally exponentially controllable (respectively, stabilizable).

Proposition 2: [1] Suppose there exists a smooth feedback control law $u = k(x,t) + v$ that makes the closed-loop system strictly contracting with rate λ in some metric $M(x,t)$ for any piecewise-continuous signal $v(t)$. Then, for all x, u, t,

$$\dot{M} + (A + BK)^T M + M(A + BK) < -2\lambda M, \tag{9}$$

where $K = \frac{\partial k}{\partial x}$, and for $\delta_x \neq 0$, the following is true:

$$\delta_x^T MB = 0 \implies \delta_x^T (\dot{M} + (A + BK)^T M + MA + 2\lambda M)\delta_x < 0. \tag{10}$$

Theorem 1: If there exists a uniformly bounded metric $M(x,t)$, i.e., $\alpha_1 I \preceq M(x,t) \preceq \alpha_2 I$, for which (10) holds for all $\delta \neq 0, x, u, t$, then System (7) is

1) universally exponentially open-loop controllable;
2) universally exponentially stabilizable via sampled-data feedback with arbitrary sample times;
3) universally exponentially stabilizable via continuous feedback defined almost everywhere, and everywhere in a neighborhood of the target trajectory;

all with rate λ and overshoot $R = \sqrt{\alpha_1/\alpha_2}$. We refer to a metric satisfying the conditions of this theorem as a CCM for the system (7).

Given a CCM, Lemma 2 in [1] establishes the existence of a differential feedback controller $\delta u = k_\delta(x, \delta x, u, t)$ that achieves closed-loop exponential stabilization of the differential dynamics (8) along all solutions:

$$\begin{aligned} \frac{d}{dt}\left(\delta x^T M \delta x\right) &= \delta x^T \dot{M} \delta x + 2\delta x^T M (A\delta x + Bk_\delta) \\ &< -2\lambda \delta x^T M \delta x \end{aligned} \tag{11}$$

and furthermore, is path-integrable, so that for any smooth path $c \in \Gamma$ and any $u^* \in \mathbb{R}^m$ and $t > 0$, integrating δu along the path c will construct a feedback controller.

Remark 2: The above description is a simplified version of the discussion in [1] below Theorem 1 in that paper. The key idea is that the path-integratability requirement is a weaker condition than k_δ being completely integrable, which is important for the convex conditions for controller synthesis.

Remark 3: Note that the Euclidean norm exponential convergence with overshoot rate stated in Theorem 1 is only achieved when the controller is constructed by integrating along the geodesic. Although the above argument only requires integratability over any smooth path, to guarantee exponential (or asymptotic) convergence, it is essential to find the geodesic and construct the controller by integrating the differential controller over the geodesic.

Remark 4: Note that the contraction conditions for a control-affine system and an autonomous systems are different in two aspects: In (9), K enters the condition analogous to linear feedback control altering closed-loop system in a pole-placement manner; In (10), we can see that only in the directions without any control authority $\delta x^T MB = 0$, the system itself (including K effect on closed-loop dynamics) must be contracting.

D. Feedback Differential Controller

One of the key benefits of contraction analysis, as stated in the introduction, is that the differential dynamics (8) is a linear time-varying system, which is easier to analyze and control compared to nonlinear systems (7). State feedback control is one of the most celebrated results in linear system control. The philosophy is simple: once the controller is in the form of state feedback, the question becomes how to design the gain K such that the closed-loop system is stable.

Similarly, we now consider a differential feedback controller $\delta_u = K(x,t)\delta_x$. By plugging the differential feedback controller back to (11), it is not difficult to see that now the condition is reduced to some matrix inequality as $k_\delta = K(x,t)\delta_x$, without directly involving $k_\delta, \delta x$. To further simplify the condition, we eliminate the dependence of $A(x,u,t)$ on u in (8) by the following assumption.

Assumption 1: Vector fields $b_i, i = 1, 2, ..., m$ are killing fields for the metric M, i.e.,

$$\forall i = 1, ..., m, \quad b_i \frac{\partial M}{\partial x} + \frac{\partial b_i}{\partial x}^T M + M \frac{\partial b_i}{\partial x} = 0$$

In particular, if B is of the form $[0, I]^T$, then the above assumption states that M must not depend on the last m state variables.

Under Assumption 1, a sufficient condition for system to be universally exponentially stablizable can be derived as follows:

There exists a scalar function $\rho(x,t) \geq 0$, such that for all x, t,

$$\dot{M} + \frac{\partial f}{\partial x}^T M + M \frac{\partial f}{\partial x} - \rho MBB^T M + 2\lambda M < 0 \tag{12}$$

One can then construct the differential gain $K(x,t) = -\frac{1}{2}\rho(x,t)B(x,t)^T M(x,t)$, which makes the closed-loop system satisfies (9). Furthermore, since $K(x,t)$ is only a function of x (not u compared to an open-loop controller, details in [1]), it is always path-integrable.

E. Dual Metrics and Convex Conditions

It is shown in [1] that using a change of variables, the condition in (12) can be transformed into a LMI representation, which makes the search for a CCM convex.

Combining the condition in Assumption 1 and (12), the convex conditions for the search of a CCM can be stated as:

1) Vector fields $b_i, i = 1,2,...,m$ are killing fields for the metric W, i.e.,

$$\forall i = 1,...,m, \quad \frac{\partial W}{\partial x} b_i - W\frac{\partial b_i}{\partial x}^T W - W\frac{\partial b_i}{\partial x} = 0 \quad (13)$$

2) There exists a scalar function $\rho(x,t) > 0$ and a uniformlly positive definite matrix that satisfies: $\forall x,t$

$$\exists \beta_1, \beta_2 > 0, \quad \beta_1 I \leq W(x,t) \leq \beta_2 I \quad (14)$$

$$-\dot{W} + \frac{\partial f}{\partial x}W + W\frac{\partial f}{\partial x}^T - \rho BB^T + 2\lambda W < 0 \quad (15)$$

Remark 5: The matrix W is the dual matrix of M, which satisfies $W(x,t) = M^{-1}(x,t)$. Note that the killing field condition (13) is now a linear condition of $W(x,t)$.

Suppose we can find a matrix function $W(x,t)$ that satisfies (13)-(15), then $M(x,t) = W^{-1}(x,t)$ is a CCM, and the following controller can track the target trajectory (x^*, u^*) in a exponentially converging manner.

$$u = u^* + \int_0^1 -\frac{1}{2}\rho B^T W^{-1}\delta x ds \quad (16)$$

IV. Methods

In Section III above, we provide a skeleton of theoretical results that derive a set of sufficient conditions that allow us to search for a control contraction metric (CCM) using only convex conditions and design a controller using differential feedback laws. In this section, we describe the specific methods and procedures used in this project to develop examples for contraction analysis-based trajectory stabilization, which includes two major components: (1) Search for contraction metrics using SOS programming (2) Feedback (Tracking) controller synthesis.

A. Search for Contraction Metrics using SOS Programming

The search for contraction metrics is essentially finding uniformly positive definite matrix function $M(x,t)$ or $W(x,t)$ that satisfies linear matrix inequalities. For autonomous systems (2), the expressions in (4) and (5) are both requiring some matrices to be positive definite.

For control-affine systems (7), with the help of differential feedback control design, we are able to arrive at a stronger set of conditions that are convex in both $W(x,t)$ and $\rho(x,t)$. In addition, by using $\delta u = -\frac{1}{2}\rho B^T W^{-1}\delta_x$, the final conditions are independent of δx thus also in the form of either linear constraint (13) or requirement that some matrices are positive definie (14) (15).

Analogous to the Sum-of-squares polynomial idea, SOS matrix introduced in [8] is used to certify positive definiteness, which reduces an infinite-dimensional search to a finite-dimensional one by polynomial-basis parameterization.

1) Sum-of-squares matrix:

Definition 2: [3] Consider a symmetric matrix with polynomial entries $S(x) \in \mathbb{R}[x]^{m\times m}$, and let $y = [y_1, y_2, ..., y_m]^T$ be a vector of new indeterminates. Then $S(x)$ is a sum of squares matrix if the scalar polynomial $y^T S(x) y$ is a sum of squares in $\mathbb{R}[x,y]$. In addition, a matrix $S(x)$ is strictly SOS if $S(x) - \varepsilon I$ is a SOS matrix for some $\varepsilon > 0$.

To clarify the notations, $\mathbb{R}[x]^{m\times m}$ refers to a $m \times m$ matrix where every entry is a polynomial of x, where x is a vector of indeterminates.

Remark 6: Similar to SOS polynomials, the constraint that a matrix is strictly SOS is a stronger condition than stating a matrix is positive definite. There are matrices that are not (strictly) SOS but may very well be still positive definite.

The benefits of using matrix SOS here is that combined with the convex conditions for CCM, we can now formulate the search for CCM as a convex optimization problem.

2) Algorithmic procedure: Let us use an example [3] to describe the algorithmic procedure to certify a 2-by-2 matrix function $M(x), x \in \mathbb{R}^2$ is positive definite:

1) $M(x)$ can be parametrized as

$$M(x) = \begin{bmatrix} \sum a_{ij} x_1^i x_2^j & \sum b_{ij} x_1^i x_2^j \\ \sum b_{ij} x_1^i x_2^j & \sum c_{ij} x_1^i x_2^j \end{bmatrix}, \quad 0 \leq i+j \leq 2$$

   In drake, this is implemented as three free polynomials of x, in which two are the diagonal terms, and the other one fits both off-diagonal terms to ensure $M(x)$ is a symmetric matrix.
2) Define new indeterminate $y \in \mathbb{R}^2$
3) Add constraint $y^T M(x) y - \varepsilon y^T y$ is an SOS polynomial

Note that the other convex conditions for CCM search can also be implemented in this way. As an example, we can compute the polynomial matrix $R(x)$ in (6) symbolically in drake, and then add constraint $-y^T R(x) y - \varepsilon y^T y$ is an SOS polynomial.

Now we formalize the search for the contraction metrics for both the autonomous system (2) and the control-affine one (7) as convex optimization problems. For simplicity, we reduce $M(x,t)$ to $M(x)$, removing explicit dependence on time. This is reasonable because many dynamical systems do not depend on time explicitly. For this class of systems, $M(x)$ suffices.

For system (2):

$$\text{find } M(x) \quad (17a)$$

$$\text{s.t. } \forall x, y \in \mathbb{R}^n,\ y^T(M(x) - \varepsilon I)y \text{ is SOS} \quad (17b)$$

$$y^T(-R(x) - \varepsilon I)y \text{ is SOS} \quad (17c)$$

$$R(x) = \dot{M} + \frac{\partial f}{\partial x}^T M + M\frac{\partial f}{\partial x} \quad (17d)$$

For system (7):

$$\min_{W(x),\rho(x)} c(\beta_1,\beta_2,W) \text{ s.t. } \forall x,y\in\mathbb{R}^n, \tag{18a}$$

$$\rho(x)-\varepsilon \text{ is SOS} \tag{18b}$$

$$y^T(W(x)-\beta_1 I)y \text{ is SOS} \tag{18c}$$

$$y^T(-W(x)+\beta_2 I)y \text{ is SOS} \tag{18d}$$

$$y^T(-R(x)-\varepsilon I)y \text{ is SOS} \tag{18e}$$

$$R(x) = -\dot{W}+\frac{\partial f}{\partial x}W+W\frac{\partial f}{\partial x}^T-\rho BB^T+2\lambda W \tag{18f}$$

$$\forall i=1,...,m,\quad \frac{\partial W}{\partial x}b_i-W\frac{\partial b_i}{\partial x}^T W-W\frac{\partial b_i}{\partial x}=0 \tag{18g}$$

where $c(\beta_1,\beta_2,W)$ is implemented as a linear cost function to either restrict the size of the coefficients of polynomials in $W(x)$, e.g., $c(\beta_1)=-\beta_1$ encourages choice of larger W and avoids numerically small coefficients observed in the simulations. In addition, $c(\beta_1,\beta_2,W)$ can also be defined as the trace of $W(x)$ to encourage sparsity, hence reducing the complexity of expressions of $R(x)$.

Note that the decision variables here are written as $W(x),\rho(x),M(x)$ for simplicity, but the real decision variables should be the coefficients of polynomials in these variables.

B. Compute CCM-based tracking controller

After we solve (18) and successfully find a contraction metric $M(x)=W^{-1}(x)$, we still need to compute the tracking controller in (16). Recall that in Section III.C (Remark 3), the tracking controller can only guarantee exponential/asymptotic convergence performance to the target trajectory (x^*,u^*) if the path-integral (16) is evaluated along the geodesic γ, not an arbitrary smooth curve $c\in\Gamma$. Therefore, in order to compute the tracking controller and achieves exponential/asymptotic tracking objective, we have to first compute the geodesic in a numerical algorithm.

By definition, the geodesic is the minimum-length smooth curve, i.e.,

$$\gamma = argmin_{c\in\Gamma} L(c,t),\quad L(c,t)=\int_0^1\sqrt{\delta x^T M(x)\delta x}ds.$$

Numerically, we can use line segments between sampled points to approximate a smooth curve, and search for the minimum-length path under found Riemannian metric $M(x)$. The first challenge to formulating this optimization problem is that the sqaure root sum objective function is not differentiable, which prohibits most solvers using gradient-descent methods (e.g., "ipopt"). Luckily, for computing geodesics, the discussion in Section III.A shows that the geodesic has to be a global minimizer of the associated Riemanian energy, i.e.,

$$\gamma = argmin_{c\in\Gamma} E(c,t),\quad E(c,t)=\int_0^1\delta x^T M(x)\delta x ds,$$

thus transforming to objective into a quadratic one. Now we have a sampled-based approximation optimization problem: At every sampled time instant t:

$$\min_{\{s_i\}_{i=0}^N}\sum_{i=0}^{N-1}(s_{i+1}-s_i)^T M(s_i)(s_{i+1}-s_i) \tag{19a}$$

$$\text{s.t. } s_i\in\mathbb{R}^n \tag{19b}$$

$$s_0=x^*(t),\quad s_N=x(t) \tag{19c}$$

Essentially, at time instant t, we are finding $N-1$ points $\{s_i\}_{i=1}^{N-1}$ between the true system state $x(t)=s_N$ and the target trajectory state $x^*(t)=s_0$ such that the sum of each segment's squared norm under the Riemannian metric evaluated at the start of each segment is minimized.

Note that even with the approximation in (19), the finite-dimensional optimization problem is still a non-convex one, as $M(s_i)$ in general will be a polynomial hence the objective function will be non-convex in s_i.

Remark 7: The line segment approximation to a smooth curve is far from ideal even if N is very large number. The assumption of $M(x)$ being constant within each segment is not really reasonable especially in a nonlinear system setting. There are more advanced ways to search for a geodesic in planning algorithm literature.

Remark 8: For constant Riemannian metric M, the geodesic is always a straight line between $x^*(t)$ and $x(t)$. In that case, we don't need to run the optimization problem to find the geodesic.

Once we find such sampled points $\{s_i\}_{i=0}^N$, the tracking controller integration along the geodesic in (16) can be approximated by doing the following summation:

$$u(t)=u^*(t)-\frac{1}{2}\sum_{i=0}^{N-1}\rho(s_i)B^T W(s_i)^{-1}(s_{i+1}-s_i) \tag{20}$$

In the case where $M(x)=M$ is a constant matrix and the geodesic is a straight line, the tracking controller can be slightly simplified as follows. Note that the need to numerically integrate $\rho(x)$ still persists, but there is no need to search for a geodesic, therefore, equal-distanced sampled points on the straight line suffices.

$$u(t)=u^*(t)-\frac{1}{2}\sum_{i=0}^{N-1}\rho(s_i)B^T W^{-1}(s_{i+1}-s_i) \tag{21}$$

where $s_i=(x(t)-x^*(t))\frac{i}{N}+x(t)$.

V. Simulation Results

In this section, we present the simulation results that apply the notion of matrix SOS to certify contraction conditions, search for contraction metrics and design tracking controllers. The experimental exploration consists of two parts: (1) Search for contraction metrics for autonomous systems (2) and use found contraction metrics for stability analysis (2) Tracking controller design based on control contraction metrics.

A. Contraction Metrics for Stability Analysis

Let us focus on autonomous systems in (2) first and implement an approach that use contraction metrics to verify "incremental stability", i.e., exponential convergence of two

arbitrary trajectories. We experiment on a linear system and a nonlinear Moore-Greitzer model of a jet engine, both of which are stable.

In this section, we use pydrake and Mosek to solve the SOS programming problem (17) for contraction metric $M(x)$.

First, let's consider a simple stable linear system

$$\dot{x} = Ax, \quad x = [q, \dot{q}]^T, \quad A = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$

For linear systems, contraction metrics are always constant, therefore, we parameterize every element of M as a degree 0 polynomial of indeterminate x. In addition, when contraction metrics are independent of x, the geodesics should be straight lines between two trajectories.

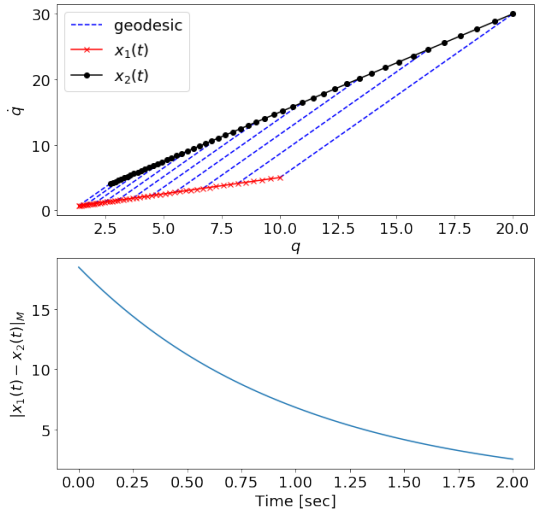

Fig. 1: Results for linear autonomous system.

Next, we consider the Moore-Greitzer jet engine model used in [3]. The system has one real-value equilibrium at the origin and it is stable.

$$\begin{bmatrix} \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} -\psi - \frac{3}{2}\phi^2 - \frac{1}{2}\phi^3 \\ 3\phi - \psi \end{bmatrix}$$

Through the experiments, it was found that using 4-th order polynomials for elements $M(x)$ is adequate. For spatially-varying contraction metrics, the geodesics computation follows the optimization problem in (19).

The initial conditions for two trajectories are chosen as $[1,2]$ and $[4,4]$. Using $N = 20$ (19 sampled points between each $x_1(t)$ and $x_2(t)$), the resulting geodesics are straight lines for $t = 4, 5, ..., 20$ [sec] as shown in Fig. 2.

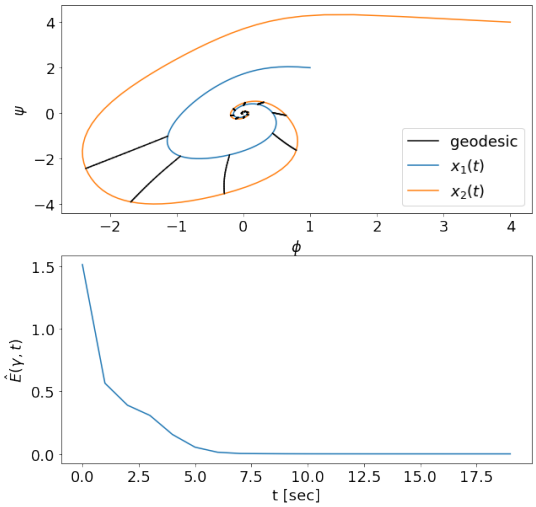

Fig. 2: Results for the jet engine model.

Although the estimated Riemannian energy of the found "geodesic", $\hat{E}(\gamma,t)$, is indeed exponentially decaying as shown in Fig. 2, the geodesic approximations do not seem to be accurate.

Note that the geodesics for $t = 0,1,2,3$ [sec] are not plotted in Fig. 2 for visual clarity. Those lines do not seem to well approximate the geodesics, which are expected to be a smooth curve. As an example, at $t = 0$ [sec], with $N = 20$, the geodesics computed by (19) can be visualized in Fig. 3.

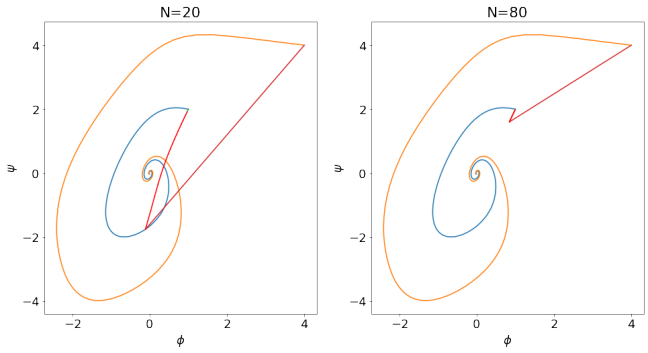

Fig. 3: Geodesic visualization with $N = 20, 80$.

One might expect that increasing number of sampling points would result in better approximation. Fig. 3 also presents the geodesic visualization with $N = 80$, which still maintains the two line segment non-smooth shape.

In addition, with increased sampling points $N = 80$, the computation time for the optimization problem (19) also increases. As discussed in Section IV.B under (19), the problem is non-convex and there is no guarantee for finding global minimizers. In the experiments, solving (19) is indeed challenging, with solver failure observed for some time instants t with $N = 80$.

Several techniques were attempted to resolve the issue but unsuccessful. Although the found geodesics shape do not change significantly with any of these techniques, we list them for a book-keeping purpose:

- Set an initial guess for sample points location to be equally-distanced points on the straight line connecting $x_1(t), x_2(t)$. The local optimizer found by (19) should at least be bounded by the initial guess cost.
- Evaluate the geodesic at the midpoint $(s_i + s_{i+1})/2$, instead of s_i to avoid sampling at points with very small $M(s_i)$ values.

B. CCM-Based Tracking Controller Synthesis

Now that we have demonstrated existence of contraction metrics is sufficient for incremental stability, we continue to explore controller synthesis through convex programming following the discussion in Section III.C-E and Section IV.B.

We consider three different case studies including two polynomial systems and one linear double-integrator. For the first polynomial system, we use a known continuous stabilizing controller to generate the target trajectory x^*, u^*. For the double integrator, the reference control signal is the analytical optimal control signal derived in the textbook. For the second polynomial system, constructing a stabilizing controller is non-trivial, therefore, we implement a piecewise constant optimal controller through trajectory optimization to generate the target trajectory.

Currently, it seems that CCM-based tracking controller in (16) cannot achieve exponential tracking performance with respect to the target trajectory for double integrator and the second polynomial system with optimal controller. My hypothesis of the reason behind this is the discontinuity in the controller used to generate target trajectory u^*. We will discuss this in more details after presenting the results. Note that in this section we do not visualize geodesics as the computation challenge has been well explained in the last one.

1) Case 1: Polynomial system with continuous control: The system in consideration is

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}-x_1 - x_1^3 + x_2^2\\ 0\end{bmatrix} + \begin{bmatrix}0\\1\end{bmatrix} u$$

Let us consider a Lyapunov function $V = \frac{1}{2}x^T x$ and controller $u^* = -x_1 x_2 - x_2$, we can show that u^* is stabilizing:

$$\begin{aligned}\dot{V} &= [x_1, x_2]\begin{bmatrix}-x_1 - x_1^3 + x_2^2\\ -x_1 x_2 - x_2\end{bmatrix}\\ &= -x_1^2 - x_2^4 - x_1^2 \leq 0\end{aligned}$$

Since V is radially unbounded and positive definite, and that $\dot{V}$ is negative definite, the above argument shows that u^* can globally stabilize a trajectory starting at any point in $\mathbb{R}^2$.

The contraction metrics are parameterized as constant, independent of x, therefore, all geodesics are straight lines. The controller synthesis follows the procedure in Section IV.B, solving (18) and (21). Note that the constraint (13) is dropped because B and W are both constants. The $\rho(x,t)$ function is parameterized as a 2nd degree polynomial.

Given the target trajectory starting at $[-2,2]$ and the real trajectory starting at $[1,3]$. The CCM-based controller does achieve satisfying tracking performance, as shown in Fig. 4. The simulation is implemented with "solve_ivp" in scipy using RK45 method.

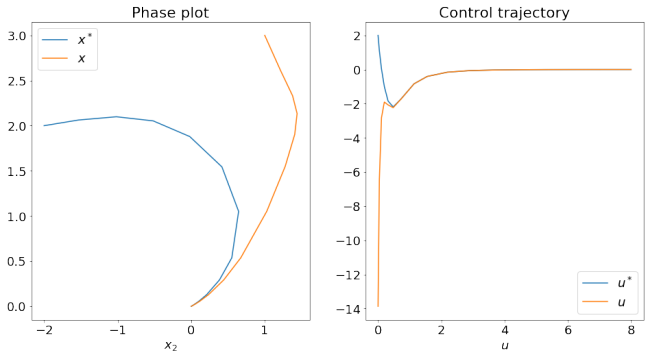

Fig. 4: Simulation results for Poly-system 1.

2) Case 2: Double Integrator with analytical optimal control: Now we consider the double integrator example,

$$\dot{x} = Ax + Bu, \quad A = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}, \quad B = \begin{bmatrix}0\\1\end{bmatrix}$$

The target trajectory is generated using the analytically optimal controller under an input limit $|u| \leq 1$ derived in the textbook. Note that this controller is generally not continuous, and can observe some switching behavior near the origin.

Since this is a linear system, both contraction metrics W and the scalar function ρ are parameterized as constant. In simulation, we concatanate the state vectors of both the target trajectory and the true trajectory, i.e., $\bar{x} = [x^{*T}, x^T]^T$. This allows us to use "solve_ivp" to mimic a continuous-time simulation.

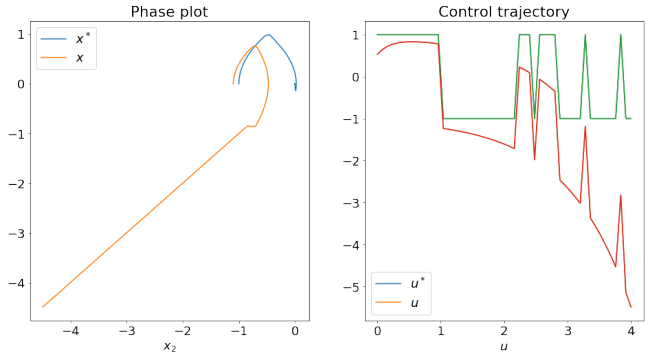

Fig. 5: Simulation results for Double Integrator.

As shown in Fig. 5, the CCM-based controller fails to track the target trajectory. More interestingly, in this case, although the target trajectory is very close to the origin, the target control u^* is still switching to try to get to the perfect origin state. The CCM-based controller seems to have increasing actuation but fails its tracking objective.

One reason behind the performance issue might be the simulation environment, i.e., the RK45 integration method we used. During simulation, it was observed that running a $t \in [0,4]$ [sec] simulation takes more than 1 minute. It seems

that the numerical integration is running into issues with the switching behavior in u^*.

Further investigation is needed, and the question of interest is how the CCM-based controller behaves if target trajectory is discontinuous or even switching.

3) Case 3: Polynomial system with sample-based optimal control: Now we consider a more complicated polynomial system with $x \in \mathbb{R}^3$ and dynamics (7) with

$$f(x) = \begin{bmatrix} -x_1 + x_3 \\ x_1^2 - x_2 - 2x_1x_3 + x_3 \\ -x_2 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Since a stabilizing controller via a Lyapunov framework is non-trivial for this system [1], we compute a sample-based piecewise constant optimal controller, using trajectory optimization and direct transcription.

$$\begin{aligned} \min_{\{x_k\},\{u_k\}} & \sum_{k=0}^{N} x_{k+1}^T Q x_{k+1} + u_k^T R u_k \\ \text{s.t. } & x_{k+1} = (f(x_k) + Bu_k)\Delta t + x_k \\ & x_0 = x^*(0) \end{aligned}$$

where $x^*(0)$ is the specified target trajectory initial condition. The optimal controller stabilizes the trajectory to the origin.

The contraction metric $W(x)$ is parameterized using 2nd order polynomials as its elements, and the scalar function $\rho(x)$ is also parameterized using 2nd order polynomial of x. The cost function in (18) is defined as $\beta_2 - \beta_1$ to reduce the coefficients of $W(x)$ while avoiding very small coefficient values. Mosek can solve for $W(x)$ and $\rho(x)$, however, the coefficients of $\rho(x)$ are very large.

$$\begin{aligned} \rho(x) = & 4514.78 - 189.03x_1 + 1.8 * x_2 - 46.52x_3 - 3.67x_1x_2 \\ & + 6.28x_1x_3 - 0.68x_2x_3 + 4460.06x_1^2 \\ & + 4463.00x_1^2 + 4453.86x_2^2 \end{aligned}$$

As a result, the CCM-based controller cannot be successfully synthesized. After the geodesic approximation is computed, the large coefficiens in $\rho(x)$ result in unbounded control signal $u(t)$, which in simulation produces "NaN" meaning the value is too large. Due to the time limit, we are unable to make this example working. One might suspect adding a uniform upper bound constraint on $\rho(x)$ can be a solution. Another reason might be the discrepancy of the model introduced by Euler discretization.

Discussion: Case 2 and 3 introduce two more complexities into the framework. Specifically, in Case 2, the target control is highly discontinuous, but at any time instant, the CCM-based controller can be constructed in a continuous-time manner; In case 3, because there is no continuous-time target controller, the CCM-based controller can only be synthesized using a discrete-time setting, in which Euler discretization can be a source of error as well.

Theoretically, these issues have been discussed in details under Theorem 1 in [1]. It seems the framework should work for sampled-data feedback controller. In addition, [1] assumes the target control $u^*(t)$ is piecewise continuous, which also seems to be able to accommodate discontinuous $u^*(t)$.

VI. Conclusions and Future Work

In this project, we have completed our learning objectives: study contraction analysis for stability analysis and controller synthesis using SOS programming. We revisited main theoretical results, and developed simulations that include working examples of stability analysis and tracking controller design. Overall, the results are encouraging in terms of providing a new perspective of incremental stability, and a fast online nonlinear control framework. Since contraction metrics can be computed offline, and the only online computation will be geodesic approximation and controller path-integral along the geodesic, which would be much faster than online optimization schemes such as MPC.

In addition, we experience issues with computing accurate approximations to geodesics, and controller performance when the target trajectory control is discontinuous in time or piecewise constant in a sampled form. These issues encourage future investigation into sampled-feedback controller synthesis following the line of work in [1]. One other way to think about this is that it might be more natural to directly adopt a discrete-time framework for contraction analysis. Very recently such a framework is proposed in [9].

Furthermore, an immediate future work would be application of the framework described in this report to robotic systems, i.e., systems with manipulator equation dynamics, following the work in [10]. One potential challenge in applying CCM to a system such as cart-pole is efficient parameterization of the monomial basis to include $\sin(x), \cos(x)$. Specifically, it involves the implementation of constraints $s^2 + c^2 = 1$ and the impact of such constraints to solving the SOS programming problem is unknown. Given more time, perhaps one might find some clues in [10].

Due to the time limit, there is one other idea that I have not yet sufficiently explored, which is the notion of local contraction and region of contraction analysis. So far, it seems all the results in literature have been focused on finding a contraction metric that provides incremental stability over the whole state space. One might wonder if a local contraction region will be helpful especially in the context of trajectory stabilization, it might not be surprising if a local CCM-based controller would outperform LQR. Analogous to the ROA estimation problem introduced in class, we can use a notion of "region of contraction". However, the difficulty is how to find an invariant set. I tested out an S-procedure implementation which asks for the largest ball for the Van der pol oscillator within which a contraction metric can be found, and the radius turns out to be around 0.85, which is much smaller than what we have seen in problem sets. Obviously, one cannot discuss region of contraction without invariant set, but Lyapunov function level set would be more natural to construct an invariant set. I am wondering if it would be interesting to study how to construct a Lyapunov function based on the differential element's energy $E(c,t)$ as they are already differential Lyapunov functions.

REFERENCES

[1] I. R. Manchester and J.-J. E. Slotine, "Control contraction metrics: Convex and intrinsic criteria for nonlinear feedback design," *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 3046–3053, 2017.

[2] W. Lohmiller and J.-J. E. Slotine, "On contraction analysis for nonlinear systems," *Automatica*, vol. 34, no. 6, pp. 683–696, 1998.

[3] E. Aylward, P. A. Parrilo, and J.-J. Slotine, "Algorithmic search for contraction metrics via sos programming," in *2006 American Control Conference*. IEEE, 2006, pp. 6–pp.

[4] I. R. Manchester and J.-J. E. Slotine, "Control contraction metrics and universal stabilizability," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 8223–8228, 2014.

[5] D. Sun, S. Jha, and C. Fan, "Learning certified control using contraction metric," *arXiv preprint arXiv:2011.12569*, 2020.

[6] H. Tsukamoto and S.-J. Chung, "Neural contraction metrics for robust estimation and control: A convex optimization approach," *IEEE Control Systems Letters*, vol. 5, no. 1, pp. 211–216, 2020.

[7] N. Rezazadeh, M. Kolarich, S. Kia, and N. Mehr, "Learning contraction policies from offline data," *IEEE Robotics and Automation Letters*, 2022.

[8] K. Gatermann and P. A. Parrilo, "Symmetry groups, semidefinite programs, and sums of squares," *Journal of Pure and Applied Algebra*, vol. 192, no. 1-3, pp. 95–128, 2004.

[9] L. Wei, R. Mccloy, and J. Bao, "Control contraction metric synthesis for discrete-time nonlinear systems," *IFAC-PapersOnLine*, vol. 54, no. 3, pp. 661–666, 2021.

[10] I. R. Manchester, J. Z. Tang, and J.-J. E. Slotine, "Unifying robot trajectory tracking with control contraction metrics," in *Robotics Research*. Springer, 2018, pp. 403–418.

APPENDIX

All the codes used to produce numerical examples are implemented using pydrake and can be found in this github repo: `https://github.com/Tangsun/6_832_final_project`.