Learning Neural Contracting Dynamics: Extended Linearization and Global Guarantees

Sean Jaffe^{1,2*}**, Alexander Davydov**^{1}**, Deniz Lapsekili**^{2}**, Ambuj K. Singh**^{2}**, and Francesco Bullo**^{1}

^{1} Center for Control, Dynamical Systems and Computation, University of California, Santa Barbara.
^{2} Department of Computer Science, University of California, Santa Barbara.

Abstract

Global stability and robustness guarantees in learned dynamical systems are essential to ensure well-behavedness of the systems in the face of uncertainty. We present Extended Linearized Contracting Dynamics (ELCD), the first neural network-based dynamical system with global contractivity guarantees in arbitrary metrics. The key feature of ELCD is a parametrization of the extended linearization of the nonlinear vector field. In its most basic form, ELCD is guaranteed to be (i) globally exponentially stable, (ii) equilibrium contracting, and (iii) globally contracting with respect to some metric. To allow for contraction with respect to more general metrics in the data space, we train diffeomorphisms between the data space and a latent space and enforce contractivity in the latent space, which ensures global contractivity in the data space. We demonstrate the performance of ELCD on the high dimensional LASA, multi-link pendulum, and Rosenbrock datasets.

1 Introduction

Due to their representation power, deep neural networks have become a popular candidate for modeling continuous-time dynamical systems of the form

$$\dot{x} = \frac{dx}{dt} = f(x), \tag{1}$$

where $f(x)$ is an unknown (autonomous) vector field governing a dynamical process. Beyond approximating the vector field f, it is desirable to ensure that the learned vector field is well-behaved. In many robotic tasks like grasping and navigation, a well-behaved system should always reach a fixed endpoint. Ideally, a learned system will still stably reach the desired endpoint even when pushed away from demonstration trajectories. Additionally, the learned system should robustly reach the desired endpoint in the face of uncertainty. In tasks such as manufacturing, animation, and human-robot interaction, functionality and safety require the learned system must smoothly follow a specific trajectory to its target.

To enforce stability guarantees, a popular approach has been to ensure that the learned dynamics admit a unique equilibrium point and have a Lyapunov function, e.g. [30]. While popular, approaches based on Lyapunov functions typically struggle to provide general robustness guarantees since global asymptotic stability does not ensure robustness guarantees in the presence of disturbances. Indeed, input-to-state stability of global asymptotically stable dynamical systems needs to be separately established, e.g., see Chapter 5 in [21].

To ensure robustness and to allow for smooth trajectory following, there has been increased interest in learning contracting dynamics from data [6]. A dynamical system is said to be contracting if any

*Corresponding author: `sjaffe@ucsb.edu` two trajectories converge to one another exponentially quickly with respect to some metric [27]. If a learned contracting system that admits a demonstration trajectory is pushed off that trajectory, it will follow a new trajectory that exponentially converges to the demonstration trajectory. Additionally, if a system is contracting, it is exponentially incrementally input-to-state stable [39]. If the system is autonomous, it also admits a unique equilibrium, is input-to-state stable, and has two Lyapunov functions that establish exponential stability of the equilibrium [9]. Since establishing contractivity globally requires satisfying a matrix partial differential inequality everywhere, prior works have focused on establishing contractivity in the neighborhood of training data [35, 36].

1.1 Related Works

Stable, but not necessarily contracting dynamics. Numerous works have aimed to learn stable dynamical systems from data including [7, 15, 22, 30, 32, 40, 45]. In [22], the authors introduce the Stable Estimator of Dynamical Systems (SEDS), which leverages a Gaussian mixture model to approximate the dynamics and enforce asymptotic stability via constraints on learnable parameters. In [30], the authors jointly learn the dynamics and a Lyapunov function for the system and project the dynamics onto the set of dynamics which enforce an exponentially decay of the Lyapunov function. This projection is done in closed-form and establishes global exponential convergence of the dynamics to the origin. In [40], the authors introduce Imitation Flow where trajectories are mapped to a latent space where states evolve in time according to a stable SDE. In [32], Euclideanizing Flows is introduced where the latent dynamics are enforced to follow natural gradient dynamics [2]. A similar approach is taken in [45] where additionally collision avoidance is considered.

Existing works on learning contracting dynamics. Learning contracting vector fields from demonstrations has attracted attention due to robustness guarantees [6, 33, 35, 37, 39]. In [33], the dynamics are defined using a Gaussian mixture model and the contraction metric is parametrized to be a symmetric matrix of polynomial functions. Convergence to an equilibrium is established using partial contraction theory [41]. In [35], the dynamics are defined via an optimization problem over a reproducing kernel Hilbert space; the dynamics are constrained to be locally contracting around the data points, i.e., there may be points in state space where the dynamics are not contracting. In [37] and [39], the authors study controlled dynamical systems of the form $\dot{x} = f(x, u)$, where u is a control input and train neural networks to find a feedback controller such that the closed-loop dynamics are approximately contracting, i.e., contractivity is not enforced but instead lack of contractivity is penalized in the cost function.

Recent work, [6], has proposed a Neural Contractive Dynamical System (NCDS) which learns a dynamical system which is explicitly contracting to a trajectory. NCDS parametrizes contracting vector fields by learning symmetric Jacobians and performing line integrals to evaluate the underlying vector field. NCDS is computationally costly because of this integration. Additionally, the Jacobian parametrization used is overly restrictive, because not all contracting vector fields have symmetric Jacobians. Constraining the vector field to have symmetric Jacobian is equivalent to enforcing that the vector field is a negative gradient flow and contracting with respect to the identity metric [42]. For scalability to higher dimensions, NCDS leverages a latent space structure where an encoder maps the data space to a lower-dimensional space and enforces contracting dynamics in this latent space. Then a decoder “projects" the latent-space dynamics to the full data space. It is then argued that on the submanifold defined by the image of the decoder, the dynamics are contracting.

1.2 Contributions

In this paper, we present a novel model for learning deep dynamics with global contraction guarantees. We refer to this model as Extended Linearized Contracting Dynamics (ELCD). To the best of our knowledge, ELCD is the first model to ensure global contraction guarantees. To facilitate the development of this model, we provide a review of contracting dynamics and extended linearization of nonlinear dynamics. Leveraging extended linearization, we factorize our vector field as $f(x) = A(x, x^*)(x - x^*)$, where x^* is the equilibrium of the dynamics. We enforce negative definiteness of the symmetric part of $A(x, x^*)$ everywhere and prove (i) global exponential stability of x^*, (ii) equilibrium contractivity of our dynamics to x^*, and (iii) using a converse contraction theorem, contractivity of the dynamics with respect to some metric.

Since negative definiteness of the symmetric part of A is not sufficient to capture all contracting dynamics, we introduce a latent space of equal dimension as the data space and learn diffeomorphisms between the data space and this latent space. The diffeomorphisms provide additional flexibility in the contraction metric and allow learning of arbitrary contracting dynamics compared to those which are solely equilibrium contracting.

Our example in Section 3.4 provides theoretical justification for why the diffeomorphism and the learned contracting model must be trained jointly. In summary, if the diffeomorphism is trained first, and transforms the data to a latent space, the model class may not be expressive enough to accurately represent the true latent dynamics. If the diffeomorphism and dynamics are trained simultaneously, this limitation is overcome. This is in contrast to [6] which trains the diffeomorphism independently as a variational autoencoder and then trains the model after on the transformed data.

Our model, ELCD, directly improves on NCDS in several ways. We parameterize the vector field directly instead of parametrizing its Jacobian. Doing so prevents us from needing to integrate the Jacobian to calculate the vector field and thus speeds up training and inference. ELCD is also more expressive than NCDS because it can represent vector fields with asymmetric Jacobians. Additionally, ELCD is guaranteed to be contracting to an arbitrary equilibrium point, either selected or learned, at all training steps. NCDS, in contrast, must learn the equilibrium point over the course of training. Additionally, NCDS learns an encoder and decoder for a lower-dimensional latent space and thus can only be contracting on the submanifold defined by the image of the decoder. From initial conditions that are not on this submanifold, NCDS may not exhibit contracting behavior. In contrast, since the latent space of ELCD is of the same dimension as the data space, we train diffeomorphisms that ensure global contractivity in the data space. ELCD exhibits better performance than NCDS at reproducing trajectories from the LASA [26], n-link Pendulum, and Rosenbrock datasets. We additionally compare against the Euclideanizing Flow [32], and Stable Deep Dynamics [30] models.

2 Preliminaries

We consider the problem of learning a dynamical system $\dot{x} = f(x)$ using a neural network that ensures that the dynamics are contracting in some metric. Going forward, we denote by $Df(x) = \frac{\partial f}{\partial x}(x)$ the Jacobian of f evaluated at x. To this end, we define what it means for a dynamical system to be contracting.

Definition 1. *A contracting dynamical system is one for which any two trajectories converge exponentially quickly. From [27], for a continuously differentiable map $f : \mathbb{R}^d \to \mathbb{R}^d$, the dynamical system $\dot{x} = f(x)$ is contracting with rate $c > 0$ if there exists a continuously-differentiable matrix-valued map $M : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ and two constants $a_0, a_1 > 0$ such that for all $x \in \mathbb{R}^n$, $M(x) = M(x)^\top$ and $a_0 I_d \succeq M(x) \succeq a_1 I_d$ and additionally satisfies for all x*

$$M(x)Df(x) + Df(x)^\top M(x) + \dot{M}(x) \preceq -2cM(x). \tag{2}$$

The map M is called the *contraction metric* and the notation $\dot{M}(x)$ is shorthand for the $n \times n$ matrix whose (i, j) entry is $\dot{M}(x)_{ij} = \nabla M_{ij}(x)^\top f(x)$. A central result in contraction theory is that any dynamical system $\dot{x} = f(x)$ satisfying (2) has any two trajectories converging to one another exponentially quickly [27, 29, 39]. Specifically, there exists $L \geq 1$ such that any two trajectories x_1, x_2 of the dynamical system satisfy

$$\|x_1(t) - x_2(t)\|_2 \leq L e^{-ct} \|x_1(0) - x_2(0)\|. \tag{3}$$

In other words, contractivity establishes exponential *incremental* stability [4].

We note that any contracting dynamical system enjoys numerous useful properties including the existence of a unique exponentially stable equilibrium and exponential input-to-state stability when perturbed by a disturbance. We refer to [27] for more useful properties of contracting dynamical systems and [9] for a recent monograph on the subject.

The problem under consideration is as follows: given a set of demonstrations $\mathcal{D} = \{(x_i, \dot{x}_i)\}_{i=1}^n$ consisting of a set of n state, $x_i \in \mathbb{R}^d$, and velocity, $\dot{x}_i \in \mathbb{R}^d$, pairs, we aim to learn a neural network $f(x_i) = \dot{x}_i$ that parametrizes a globally contracting dynamical system with equilibrium point x^*, such that $f(x^*) = 0$. In essence, this task requires learning both the vector field and the contraction metric, M such that they jointly satisfy (2).

2.1 Contracting Linear Systems

Suppose we assumed that the dynamical system we aimed to learn was linear, i.e., $\dot{x} = Ax$ for some matrix $A \in \mathbb{R}^{d \times d}$ and we wanted to find a contraction metric M which we postulate to be constant $M(x) := M$ for all x. The contraction condition (2) then reads

$$M(A + cI_n) + (A + cI_n)^\top M \preceq 0, \tag{4}$$

which implies that A has all eigenvalues with $\mathrm{Re}(\lambda(A)) \leq -c$, see Theorem 8.2 in [18]. In other words, for linear systems, contractivity is equivalent to stability.

Although the condition (4) is convex in M at fixed A, the task of learning both (A, M) simultaneously from data is not (jointly) convex. Instead, different methods must be employed such as alternating minimization. A similar argument is made in [30] in the context of stability and here we show that the same is true of contractivity.

2.2 Contractivity and Exponential Stability for Nonlinear Systems

In the case of nonlinear systems, contractivity is not equivalent to asymptotic stability. Indeed, asymptotic stability requires finding a continuously differentiable Lyapunov function $V : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ which is positive everywhere except at the equilibrium, x^*, and satisfies the decay condition

$$\nabla V(x)^\top f(x) < 0, \quad \forall x \in \mathbb{R}^d \setminus \{x^*\}. \tag{5}$$

One advantage of learning asymptotically stable dynamics compared to contracting ones is that the function V is scalar-valued, while the contraction metric M is matrix-valued. A disadvantage of learning asymptotically stable dynamics compared to contracting ones is that we are required to know the location of the equilibrium point beforehand and no robustness property of the learned dynamics is automatically enforced. To this end, there are works in the literature that enforce an equilibrium point at the origin and learn the dynamics and/or the Lyapunov function under this assumption [30]. In the case of contractivity, existence and uniqueness of an equilibrium point is implied by the condition (2) and it can be directly parametrized to best suit the data.

3 Methods

3.1 Motivation

The motivation for our approach comes from the well-known mean-value theorem for vector-valued mappings, which we highlight here.

Lemma 2 (Mean-value theorem (Proposition 2.4.7 in [1]))**.** *Let $f : \mathbb{R}^d \to \mathbb{R}^d$ be continuously differentiable. Then for every $x, y \in \mathbb{R}^d$,*

$$f(x) - f(y) = \Big(\int_0^1 Df(\tau x + (1-\tau)y)d\tau\Big)(x - y). \tag{6}$$

If $y = x^*$ satisfies $f(x^*) = 0$, then the continuously differentiable map f admits the factorization

$$f(x) = A(x, x^*)(x - x^*), \quad \text{where} \tag{7}$$

$$A(x, x^*) = \int_0^1 Df(\tau x + (1-\tau)x^*)d\tau. \tag{8}$$

This factorization is referred to as *extended linearization* in analogy to standard linearization, i.e., for x close to x^*, $f(x) \approx Df(x^*)(x - x^*)$. We remark that when $d \geq 2$, extended linearization is not unique, and for a given f, there may exist several A such that $f(x) = A(x, x^*)(x - x^*)$. In other words, (8) showcases one valid choice for A such that this factorization holds. Indeed, this nonuniqueness has been leveraged in some prior works, e.g. Section 3.1.3 in [39], to yield less conservative contractivity conditions.

Since we know that a contracting vector field admits a unique equilibrium point, x^*, we restrict our attention to learning a matrix-valued mapping $A : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d \times d}$ and ensuring that this mapping has enough structure so that the overall vector field satisfies the contraction condition (2) for some contraction metric M. This task is nontrivial since $f(x) = A(x, x^*)(x - x^*)$ implies that

$$Df(x) = A(x, x^*) + \frac{\partial A}{\partial x}(x, x^*)(x - x^*), \tag{9}$$

where $\frac{\partial A}{\partial x} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d \times d \times d}$ is a third-order tensor-valued mapping. In what follows, we will show that a more simple condition will imply contractivity of the vector field. Namely, negative definiteness of the symmetric part of $A(x, x^*)$ will be sufficient for contractivity of the dynamical system $\dot{x} = A(x, x^*)(x - x^*)$.

3.2 Parametrization of $A(x, x^*)$

We show a simple example of our model, the Extended Linearized contracting Dynamics (ELCD). Let $x \in \mathbb{R}^d$ be the state variable and $f : \mathbb{R}^d \to \mathbb{R}^d$ be a vector field with equilibrium point x^*. As indicated, we parametrize our vector field by its extended linearization

$$\dot{x} = f(x) = A(x, x^*)(x - x^*), \tag{10}$$

where now

$$\begin{aligned} A(x, x^*) = &-P_s(x, x^*)^\top P_s(x, x^*) \\ &+ P_a(x, x^*) - P_a(x, x^*)^\top - \alpha I_d \end{aligned} \tag{11}$$

$P_s, P_a : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d \times d}$ are neural networks, I_d is the d-dimensional identity matrix, and $\alpha > 0$ is a constant scalar. Note that the symmetric part of $A(x, x^*)$ is negative definite since

$$\frac{A(x, x^*) + A(x, x^*)^\top}{2} = -P_s(x, x^*)^\top P_s(x, x^*) - \alpha I_d$$

and $P_s(x, x^*)^\top P_s(x, x^*)$ is guaranteed to be positive semidefinite.

We prove that a vector field parametrized this way is guaranteed to be (i) globally exponentially stable, (ii) equilibrium contracting as defined in [10], and (iii) contracting in some metric. The key tools are partial contraction theory [41] and a converse contraction theorem.

Theorem 3 (Equilibrium Contraction and Global Exponential Stability)**.** *Suppose the dynamical system $\dot{x} = f(x)$ is parametrized as $f(x) = A(x, x^*)(x - x^*)$ where $A(x)$ is as in (11). Then any trajectory $x(t)$ of the dynamical system satisfies*

$$\|x(t) - x^*\|_2 \leq e^{-\alpha t}\|x(0) - x^*\|_2. \tag{12}$$

Proof. We use the method of partial contraction as in [41]. Let $x(t)$ be a solution to $\dot{x}(t) = A(x(t), x^\star)(x(t) - x^*)$ with initial condition $x(0) = x_0$. Then, define the time-varying virtual system

$$\dot{y}(t) = A(x(t), x^*)(y(t) - x^*). \tag{13}$$

We will establish that this virtual system is contracting in the identity metric, i.e., (2) is satisfied with $M(x) = I_d$. We see that the Jacobian for this virtual system is simply $A(x(t), x^*)$ and

$$\begin{aligned} A(x(t), x^*) + A(x(t), x^*)^\top &= -2P_s(x(t), x^*)^\top P_s(x(t), x^*) - 2\alpha I_d \\ &\preceq -2\alpha I_d. \end{aligned}$$

In other words, in view of (3), and since $M(x) = I_d$, any two solution trajectories $y_1(t)$ and $y_2(t)$ of the virtual system satisfy

$$\|y_1(t) - y_2(t)\|_2 \leq e^{-\alpha t}\|y_1(0) - y_2(0)\|_2.$$

Note that we can pick one trajectory to be $y_1(t) = x^*$ for all t and we can pick $y_2(t) = x(t)$. Since x_0 was arbitrary, this argument establishes the claim. □

Clearly, the bound (12) implies exponential convergence of trajectories of the dynamical system (10) to x^*. Moreover, this bound exactly characterizes equilibrium contractivity, as was defined in [10]. Notably, using the language of logarithmic norms, Theorem 33 in [10] establishes a similar result to Theorem 3 without invoking a virtual system.

Note that although Theorem 3 establishes global exponential stability and equilibrium contraction, it does not establish global contractivity. Indeed, the contractivity condition (2) is not guaranteed to hold with $M(x) = I_d$ without further assumptions on the structure of $A(x, x^*)$ in (10). Remarkably, however, due to a converse contraction theorem of Giesl, Hafstein, and Mehrabinezhad, it turns out that one can construct a state-dependent M such that the dynamics are contracting. Specifically, since trajectories of the dynamical system satisfy the bound (12), for any matrix-valued mapping $C : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ with $C(x) = C(x)^\top \succ 0$ for all x, the matrix PDE

$$M(x)Df(x) + Df(x)^\top M(x) + \dot{M}(x) = -C(x) \tag{14}$$

admits a unique solution for each x [16]. In other words, the unique M solving this matrix PDE serves as the contraction metric and will satisfy (2) with suitable choice for c. The following proposition provides the explicit solution for M in terms of the solution to the matrix PDE.

Proposition 4 (Theorem 2.8 in [16])**.** *Let $C : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ be smooth and have $C(x) = C(x)^\top$ be a positive definite matrix at each x. Then $M : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ given by the formula*

$$M(x) = \int_0^\infty \psi(\tau, x)^\top C(\phi(\tau, x))\psi(\tau, x)d\tau, \tag{15}$$

is a contraction metric for the dynamical system (10) on any compact subset containing x^*, where $\tau \mapsto \phi(\tau, x)$ is the solution to the ODE with $\phi(0, x) = x$ and $\tau \mapsto \psi(\tau, x)$ is the matrix-valued solution to

$$\dot{Y} = Df(\phi(t, x))Y, \quad Y(0) = I_d. \tag{16}$$

While it is challenging, in general, to compute the metric (15), numerical considerations for approximating it arbitrarily closely are presented in [17]. In practice, one would select $C(x) = I_d$ and evaluate all integrals numerically. For ELCD, unless one directly needs to know the contraction metric for application purposes, it is not required to compute the contraction metric at any point during either training or inference.

We remark that our parametrization for A in (11) is similar to the parametrization for the Jacobian of f that was presented in [6]. There are however a few key differences. Notably, $A(x, x^*)$ may be asymmetric while the Jacobian in equation (3) in [6] is always symmetric. Notably, since the Jacobian in [6] is symmetric and negative definite, the vector field $\dot{x} = f(x)$ is a negative gradient flow, $\dot{x} = -\nabla V(x)$, for some strongly convex function V. On the contrary, our dynamics (10) can exhibit richer behaviors than negative gradient flows in view of the asymmetry in $A(x, x^*)$. Additionally, since the dynamics in [6] with their parametrization can be represented as $\dot{x} = -\nabla V(x)$ for some strongly convex V, it is guaranteed to be contracting in the identity metric, $M(x) = I_d$ [42]. On the other hand, our dynamics (10) is not necessarily contracting in the identity metric and instead is contracting in a more complex metric given in (15).

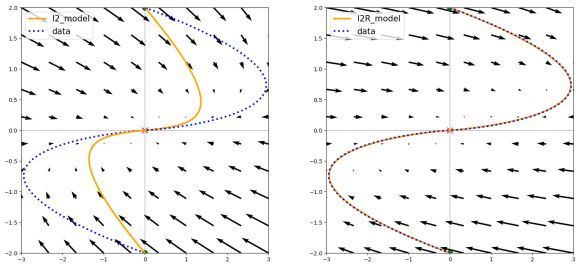
3.3 Latent Space Representation

Realistic dynamical systems and their flows including the handwritten trajectories found in the LASA dataset, are often highly-nonlinear and may not be represented in the form (10) or obey the bound (12). One solution to these challenges is to transform the system to a latent, possible lower-dimensional, space and learn an ELCD in the latent space.

Latent space learning is possible because contraction is invariant under differential coordinate changes. From Theorem 2 of [29], given a dynamical system $\dot{x} = f(x)$, $f : \mathbb{R}^d \to \mathbb{R}^d$, with f satisfying (1), the system will also be contracting under the coordinate change $z = \phi(x)$ if $\phi : \mathbb{R}^d \to \mathbb{R}^d$ is a smooth diffeomorphism. Specifically, if $\dot{x} = f(x)$ is contracting with metric M, then the system that evolves z is contracting as well with metric given by $\tilde{M}(z) = D\phi(z)^{-\top} M(z) D\phi(z)^{-1}$, where $z = \phi(x)$ and $D\phi$ is the Jacobian of the coordinate transform. In other words, We can learn vector fields that are contracting in an arbitrary metric by training a vector field f which is parametrized as (10) and using a coordinate transform ϕ.

NCDS [6], treats the coordinate transform as a Variational Autoencoder (VAE) [23]. Their training procedure consists of two steps: first training the coordinate transform with VAE training, then training the function f in the new learned coordinates.

Figure 1: The learned vector field and corresponding trajectories of an ELCD with no transform (left) and a model with a transform (right) when trained on data that is generated from a vector field that is contracting in a more general metric.



3.4 On the Interdependence of Diffeomorphism and Learned Dynamics

To demonstrate the need for a coordinate transform, we consider the task of learning a vector field that fits trajectories generated by the linear system $\dot{x} = Ax$ with $A = \begin{pmatrix} -1 & 4 \\ 0 & -1 \end{pmatrix}$. Clearly, this linear system cannot be represented in the form (10) with parametrization (11). To see this fact, we observe that $A + A^\top$ is not negative definite and thus no choice of P_s or P_a can represent this linear system. To remedy this issue, one can take the linear coordinate transform $z = \phi(x) = Px$, where $P = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix}$. Then the z-dynamics read

$$\dot{z} = PAP^{-1}z = \begin{pmatrix} -1 & 1 \\ 0 & -1 \end{pmatrix} z \tag{17}$$

and now the symmetric part of PAP^{-1} is negative definite and thus we can find suitable choices of P_s, P_a. Specifically, we can take $\alpha \in (0, 1/2)$, let $P_a(z) = \begin{pmatrix} 0 & 0 \\ 1/2 & 0 \end{pmatrix} z$, and let $P_s(z) = Qz$ where Q is the matrix square root of $\begin{pmatrix} 1 & -1/2 \\ -1/2 & 1 \end{pmatrix} - \alpha I_2$. Note that in this toy example, asymmetry is essential to exactly represent these dynamics in the latent space, z. If we used the parametrization in [6], we would not be able to represent these dynamics since they have an asymmetric Jacobian. We demonstrate this routine in Figure (1). We generate two trajectories starting at $(0, 2)$ and $(0, -2)$, and use that data to train two ELCDs, one without and one with a learned, linear transform. Figure 1(a) shows the vector field and trajectories corresponding to an ELCD with no transform. The trajectories are forced to the center sooner than the actual data as a consequence of the bound (12). Learning a coordinate transform allows the ELCD to learn a system that is contracting in an arbitrary metric and exhibit overshoot. This is demonstrated by the learned system in Figure 1(b), which perfectly matches the data.

3.5 Choice of Diffeomorphism

There are several popular neural network diffeomorphisms including coupling layers [12, 32], normalizing flows [40], $\mathcal{M}$-flow [6, 8], spline flows [14], and radial basis functions [5].

A coupling layer $\phi : \mathbb{R}^d \to \mathbb{R}^d$ consists of a neural network $\theta : \mathbb{R}^k \to \mathbb{R}^N$ with $1 < k < d$ and a neural network $g : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}$. The transform ϕ maps input $x \in \mathbb{R}^d$ to $y \in \mathbb{R}^d$ with the following procedure:

(i) Set $y_i = x_i$ for $1 \leq i \leq k$ for some $1 < k < d$.

(ii) Set $y_i = g(x_{1:k}, x_i)$ for $k \leq i \leq d$

Coupling layers are invertible by doing the above process in reverse, so long as g is invertible. A coupling layer can exhibit a wide variety of behaviors depending on the choice of g.

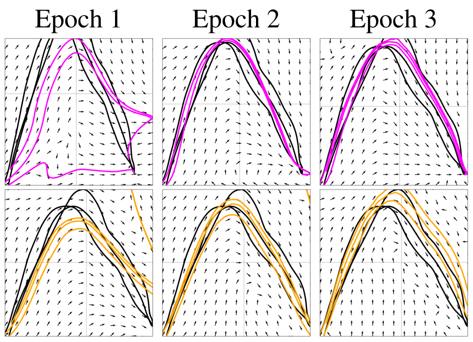
Polynomial spline curves are another diffeomorphism that are commonly used as g in coupling layers. A polynomial spline has a restricted domain which is divided into bins. A different linear, quadratic

[31], or cubic [13] polynomial covers each bin. [14] introduce rational-quadratic splines, which constructs a spline from functions that are the quotient of two quadratic polynomials.

In practice, ϕ is the composition of several of transforms. Because the coupling transform only alters some coordinates, they are often used in conjunction with random permutations. If the latent space dimension is smaller than the data space dimension [6] makes the last composite function of ϕ an 'Unpad' function, which simply removes dimensions to match the latent space. The decoder ϕ is then prepended by a 'Pad' function, which concatenates the latent variable with the necessary number of 'zeros' to match the data space.

Of course, if the latent dimension is smaller than that of the data dimension, then ϕ can not be bijective. However, [6] argue that as long as ϕ is injective and has range over the dataset, then f being contracting in the latent space implies that the learned dynamics are contracting on the submanifold defined by the image of ϕ^{-1}. In other words, NCDS cannot be globally contracting. The consequences of this are shown in Figure (2). While NCDS can learn to admit an equilibrium point when on the submanifold, trajectories that fall off the submanifold may diverge. ELCD, in contrast, is contracting to the correct equilibrium point during all phases of training.

Figure 2: Plots of the vector fields and induced trajectories learned by ELCD (Top) and NCDS (Bottom) after different training epochs. ELCD is contracting always while NCDS may admit multiple equilibrium or diverge.



3.6 Training

Our task is to simultaneously learn the contracting system $f(x)$ and the metric in which f contracts. As previously discussed, the metric is implicitly determined by $\phi(x)$. [6] uses a two-step method. First, they treat ϕ as a variational autoencoder and maximize the evidence lower bound (ELBO). They then fix the encoder and train the model to evolve the state in latent space. Note that the VAE objective is to make the encoded data to match a standard-normal distribution. Notably, VAE training does not encourage the encoder to transform the data to space in which the data corresponds to trajectories of a contracting system. If the data is not contracting in the transformed space, a contracting model will not be able to fully fit the data. This explains why, in our implementation of the two step-training, we are unable to reasonably learn the data. For further comparison with NCDS, we will train the encoder and model jointly.

4 Experiments

The code for our model and data can be found here: `https://github.com/seanjaffe1/Extended-Linearized-Contracting-Dynamics`. For all datasets we compare our method against NCDS [6], Stable Deep Dynamics (SDD) [30] and Euclideanizing Flow (EFlow) [32]. See A.2 for a detailed discussion of the methods. We report the dynamic time warping distance (DTWD) [20] (see A.1) and standard deviation between predicted and data trajectories. See A.5 for model implementation details. See also the Appendix for more details on these models.

Figure 3: Plots of the 2D LASA data. Demonstrations are in black. The learned ELCD trajectories in magenta are plotted along with the learned vector field. The vector field velocities have been normalized for visibility.

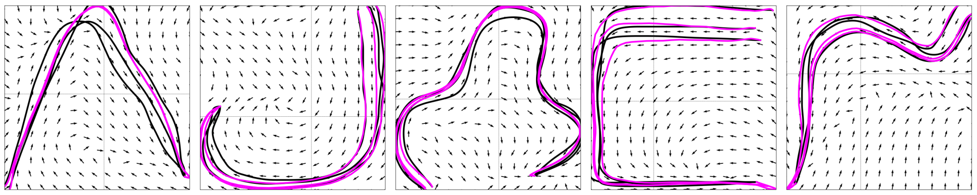


Table 1: Mean DTWD $\pm$ one standard deviation across 10 runs on LASA, multi-link pendulum, and Rosenbrock datasets

	SDD	EFlow	NCDS	ELCD
LASA-2D	0.37 ± 0.32	1.05 ± 0.25	0.59 ± 0.61	**0.12** $\pm$**0.11**
LASA-4D	2.49 ± 2.4	2.24 ± 0.12	2.19 ± 1.23	**0.80** $\pm$ **0.54**
LASA-8D	5.26 ± 0.50	2.66 ± 0.63	5.04 ± 0.77	**1.52** $\pm$ **0.61**
Pendulum-4D	0.49 ± 0.11	0.17 ± 0.01	1.35 ± 2.26	**0.03** $\pm$ **0.01**
Pendulum-8D	0.75 ± 0.08	0.33 ± 0.01	2.88 ± 0.69	**0.14** $\pm$ **0.03**
Pendulum-16D	1.86 ± 0.14	0.45 ± 0.01	1.65 ± 0.31	**0.44** $\pm$ **0.09**
Rosenbrock-8D	NaN	1.90 ± 0.16	2.74 ± 0.15	**1.22** $\pm$ **0.01**
Rosenbrock-16D	NaN	3.57 ± 0.66	3.68 ± 0.12	**2.57** $\pm$ **0.09**

4.1 Datasets

We experiment with the LASA dataset [26], which consists of 30, two-dimensional curves. We use three demonstration trajectories of each curve. As in [6], the first few initial points of each trajectory are omitted so that only the target state has zero velocity. we stack two and four LASA trajectories together to make datasets of 4 and 8-dimensional trajectories, respectively. All data is standardized to have a mean of zero and variance of one. We use in total 10 2D curves, 6 4D curves, and 6 8D curves. Some 2D-LASA trajectories and their respective trained ELCD trajectories and vector fields are visualized in Figure (3).

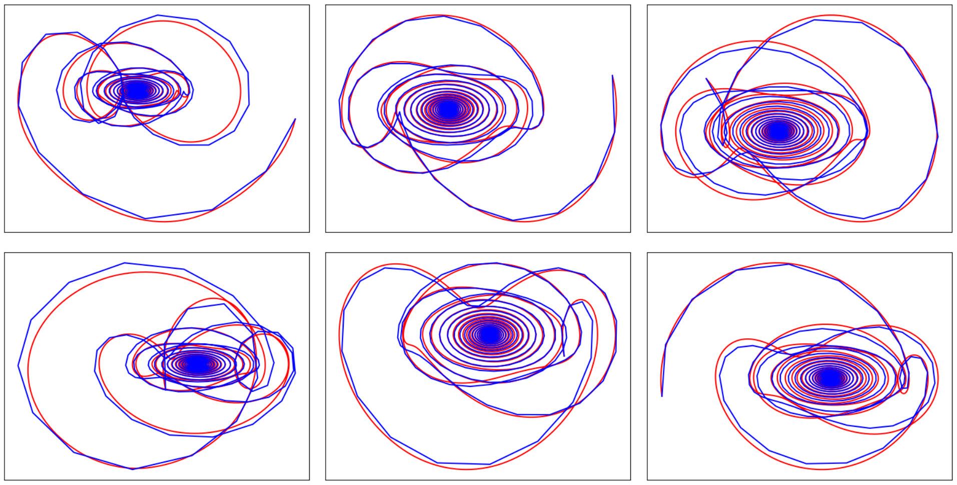
We also experiment with simulated datasets. We simulate 6 trajectories of a 2,4, and 8-link pendulum (4D, 8D, and 16D respectively) each and 4 trajectories of 8D and 16D Riemannian gradient descent dynamics on a generalization of the Rosenbrock function (see Appendix A.4). Each model is trained on all trajectories of the same dimension, and then predictions are made starting from every initial point.

4.2 Results

Table 1 presents our results. ELCD performs the best in all tasks. These results shows the benefit of the increased expressiveness allowed by the skew symmetric component of our parametrization. These benefits are more apparent in the pendulum dataset. The oscillatory behavior of the pendulum dynamics is a product of complex eigenvalues in its Jacobian. Our model's skew symmetric component is what enables it to learn the pendulum dynamics so well. A sample of demonstration and learned pendulum trajectories is shown in figure (4).

The Rosenbrock dynamics are stiff and difficult to learn. In our training of SDD, we observed lack of stability and convergence, resulting in very large DTWDs. Hence, we report NaN for the results of the SDD models on the Rosenbrock dataset. While our model performs the best, there is still room for improvement. Handling such dynamics with multiple time scales is still an open challenge.

Figure 4: Phase plots of the two link-pendulum trajectories (blue) and the trajectories produced by ELCD (red). Each column is a different trajectory. The top row is the first link and the bottom row is the second link. The x-axis is angle and the y-axis is angular velocity.



5 Limitations

Our method assumes that the underlying dynamics of the data trajectories are contracting. However, the diffeomorphism allows for a wide class of stable systems to be represented. Our method is currently implemented for trajectories that converge to the same fixed point. But, this can be overcome by manually changing the fixed point, depending on which trajectory the input data came from. Also, right now our method is limited to dynamics in $\mathbb{R}^n$, but we imagine that an extension to more general manifolds should be possible.

As we are motivated by applications in imitation learning and robotics, in this work, we are primarily interested in problems where the underlying systems should be robustly stable, especially away from training data. For this reason, we focus on learning dynamics which are guaranteed to be globally contracting and do not address learning other types of systems, such as those with multiple fixed points, limit cycles, or chaotic attractors. We imagine that extensions of ELCD could capture these richer dynamical behaviors by leveraging weaker notions of contraction including local contraction (i.e., contractivity in the region of attraction of a stable equilibrium), k-contraction (contraction of k-dimensional bodies) [43], transverse contraction [28] and contraction in the Hausdorff dimension [44]. Moreover, the notion of translation invariance mentioned could be studied using semicontraction theory [11], i.e., contraction to a subspace.

6 Conclusions

In this paper, we introduce ELCD, the first neural network-based dynamical system with global contractivity guarantees in arbitrary metrics. The main theoretical tools are extended linearization, equilibrium contraction, and a converse contraction theorem. To allow for contraction with respect to more general metrics, we use a latent space representation with dimension of the latent space equal to the dimension of the data space and train diffeomorphisms between these spaces. We highlight key advantages of ELCD compared to NCDS as introduced in [6], including global contraction guarantees, expressible parametrization, and efficient inference. We demonstrate the performance of ELCD on high-dimensional LASA datasets and simulated multi-link pendulum and Rosenbrock dynamics. Our method shows consistent performance across all datasets.

Acknowledgments and Disclosure of Funding

This work was supported in part by the NSF Graduate Research Fellowship under grant 2139319, the AFOSR under award FA9550-22-1-0059, the NSF under grant no. 2229876, the Department of Homeland Security, and by IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF or its federal agency and industry partners.

References

[1] R. Abraham, J. E. Marsden, and T. S. Ratiu. *Manifolds, Tensor Analysis, and Applications*, volume 75 of *Applied Mathematical Sciences*. Springer, 2 edition, 1988. ISBN 0387967907.

[2] S. Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998. doi:10.1162/089976698300017746.

[3] B. Amos, L. Xu, and J. Z. Kolter. Input convex neural networks. In *International Conference on Machine Learning*, 2017. URL https://proceedings.mlr.press/v70/amos17b.html.

[4] D. Angeli. A Lyapunov approach to incremental stability properties. *IEEE Transactions on Automatic Control*, 47(3):410–421, 2002. doi:10.1109/9.989067.

[5] H. Beik-Mohammadi, S. Hauberg, G. Arvanitidis, G. Neumann, and L. Rozo. Reactive motion generation on learned Riemannian manifolds. *International Journal of Robotics Research*, 42 (10):729–754, 2023. doi:10.1177/02783649231193046.

[6] H. Beik-Mohammadi, S. Hauberg, G. Arvanitidis, N. Figueroa, G. Neumann, and L. Rozo. Neural contractive dynamical systems. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=iAYIRHOYy8.

[7] P. Bevanda, M. Beier, S. Kerz, A. Lederer, S. Sosnowski, and S. Hirche. Diffeomorphically learning stable Koopman operators. *IEEE Control Systems Letters*, 6:3427–3432, 2022. doi:10.1109/LCSYS.2022.3184927.

[8] J. Brehmer and K. Cranmer. Flows for simultaneous manifold learning and density estimation. In *Advances in Neural Information Processing Systems*, pages 442–453, 2020. URL https://arxiv.org/abs/2003.13913.

[9] F. Bullo. *Contraction Theory for Dynamical Systems*. Kindle Direct Publishing, 1.1 edition, 2023. ISBN 979-8836646806. URL https://fbullo.github.io/ctds.

[10] A. Davydov, S. Jafarpour, and F. Bullo. Non-Euclidean contraction theory for robust nonlinear stability. *IEEE Transactions on Automatic Control*, 67(12):6667–6681, 2022. doi:10.1109/TAC.2022.3183966.

[11] G. De Pasquale, K. D. Smith, F. Bullo, and M. E. Valcher. Dual seminorms, ergodic coefficients, and semicontraction theory. *IEEE Transactions on Automatic Control*, 69(5):3040–3053, 2024. doi:10.1109/TAC.2023.3302788.

[12] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

[13] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios. Cubic-spline flows. In *ICML Workshop on Invertible Neural Networks and Normalizing Flows*, 2019. URL https://arxiv.org/abs/1906.02145.

[14] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios. Neural spline flows. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/abs/1906.04032.

[15] F. Fan, B. Yi, D. Rye, G. Shi, and I. R. Manchester. Learning stable Koopman embeddings. In *American Control Conference*, pages 2742–2747, 2022. doi:10.23919/ACC53348.2022.9867865.

[16] P. Giesl, S. Hafstein, and I. Mehrabinezhad. Computation and verification of contraction metrics for exponentially stable equilibria. *Journal of Computational and Applied Mathematics*, 390: 113332, 2021. doi:10.1016/j.cam.2020.113332.

[17] P. Giesl, S. Hafstein, and I. Mehrabinezhad. Contraction metrics by numerical integration and quadrature: Uniform error estimate. In *20th International Conference on Informatics in Control, Automation and Robotics*, 2023. doi:10.5220/0012183300003543.

[18] J. P. Hespanha. *Linear Systems Theory*. Princeton University Press, 2009. ISBN 0691140219.

[19] N. Jaquier and L. Rozo. High-dimensional Bayesian optimization via nested Riemannian manifolds. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2010.10904.

[20] E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, 2005. doi:10.1007/s10115-004-0154-9.

[21] H. K. Khalil. *Nonlinear Systems*. Prentice Hall, 3 edition, 2002. ISBN 0130673897.

[22] S. Mohammad Khansari-Zadeh and Aude Billard. Learning stable nonlinear dynamical systems with Gaussian mixture models. *IEEE Transactions on Robotics*, 27(5):943–957, 2011. doi:10.1109/TRO.2011.2159412.

[23] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014. URL https://arxiv.org/abs/1312.6114.

[24] L. Kozachkov, P. M. Wensing, and J.-J. Slotine. Generalization as dynamical robustness–The role of Riemannian contraction in supervised learning. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=Sb6p5mcefw.

[25] Y. Lan and I. Mezić. Linearization in the large of nonlinear systems and Koopman operator spectrum. *Physica D: Nonlinear Phenomena*, 242(1):42–53, 2013. doi:10.1016/j.physd.2012.08.017.

[26] A. Lemme, Y. Meirovitch, M. Khansari-Zadeh, T. Flash, A. Billard, and J. J. Steil. Open-source benchmarking for learned reaching motion generation in robotics. *Paladyn, Journal of Behavioral Robotics*, 6(1):30–41, 2015. doi:10.1515/pjbr-2015-0002.

[27] W. Lohmiller and J.-J. E. Slotine. On contraction analysis for non-linear systems. *Automatica*, 34(6):683–696, 1998. doi:10.1016/S0005-1098(98)00019-3.

[28] I. R. Manchester and J.-J. E. Slotine. Transverse contraction criteria for existence, stability, and robustness of a limit cycle. *Systems & Control Letters*, 63:32–38, 2014. doi:10.1016/j.sysconle.2013.10.005.

[29] I. R. Manchester and J.-J. E. Slotine. Control contraction metrics: Convex and intrinsic criteria for nonlinear feedback design. *IEEE Transactions on Automatic Control*, 62(6):3046–3053, 2017. doi:10.1109/TAC.2017.2668380.

[30] G. Manek and J. Z. Kolter. Learning stable deep dynamics models. In *Advances in Neural Information Processing Systems*, 2019. URL https://arxiv.org/pdf/2001.06116.

[31] T. Müller, B. McWilliams, F. Rousselle, M. Gross, and J. Novák. Neural importance sampling. *ACM Transactions on Graphics*, 38(5):1–19, 2019. doi:10.1145/3341156.

[32] M. A. Rana, A. Li, D. Fox, B. Boots, F. Ramos, and N. Ratliff. Euclideanizing flows: Diffeomorphic reduction for learning stable dynamical systems. In *Conference on Learning for Dynamics and Control*, pages 630–639, 2020. URL https://proceedings.mlr.press/v120/rana20a.html.

[33] H. Ravichandar, I. Salehi, and A. Dani. Learning partially contracting dynamical systems from demonstrations. In *Conference on Robot Learning*, pages 369–378, 2017. URL https://proceedings.mlr.press/v78/ravichandar17a.html.

[34] H. H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, 1960. doi:10.1093/comjnl/3.3.175.

[35] V. Sindhwani, S. Tu, and M. Khansari. Learning contracting vector fields for stable imitation learning, 2018. URL https://arxiv.org/pdf/1804.04878. Arxiv e-print.

[36] S. Singh, B. Landry, A. Majumdar, J-J. E. Slotine, and M. Pavone. Robust feedback motion planning via contraction theory. *International Journal of Robotics Research*, 42(9):655–688, 2023. doi:10.1177/02783649231186165.

[37] D. Sun, S. Jha, and C. Fan. Learning certified control using contraction metric. In *Conference on Robot Learning*, volume 155, pages 1519–1539, 2021. URL https://proceedings.mlr.press/v155/sun21b.html.

[38] T. Teshima, I. Ishikawa, K. Tojo, K. Oono, M. Ikeda, and M. Sugiyama. Coupling-based invertible neural networks are universal diffeomorphism approximators. In *Advances in Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2006.11469.

[39] H. Tsukamoto, S.-J. Chung, and J.-J. E Slotine. Contraction theory for nonlinear stability analysis and learning-based control: A tutorial overview. *Annual Reviews in Control*, 52: 135–169, 2021. doi:10.1016/j.arcontrol.2021.10.001.

[40] J. Urain, M. Ginesi, D. Tateo, and J. Peters. ImitationFlow: Learning deep stable stochastic dynamic systems by normalizing flows. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 5231–5237, 2020. doi:10.1109/iros45743.2020.9341035.

[41] W. Wang and J. J. Slotine. On partial contraction analysis for coupled nonlinear oscillators. *Biological Cybernetics*, 92(1):38–53, 2005. doi:10.1007/s00422-004-0527-x.

[42] P. M. Wensing and J.-J. E. Slotine. Beyond convexity — Contraction and global convergence of gradient descent. *PLoS One*, 15(8):1–29, 2020. doi:10.1371/journal.pone.0236661.

[43] C. Wu, I. Kanevskiy, and M. Margaliot. k-contraction: Theory and applications. *Automatica*, 136:110048, 2022. doi:10.1016/j.automatica.2021.110048.

[44] C. Wu, R. Pines, M. Margaliot, and J.-J. E. Slotine. Generalization of the multiplicative and additive compounds of square matrices and contraction in the Hausdorff dimension. *IEEE Transactions on Automatic Control*, 2022. doi:10.1109/TAC.2022.3162547.

[45] W. Zhi, T. Lai, L. Ott, and F. Ramos. Diffeomorphic transforms for generalised imitation learning. In *Learning for Dynamics and Control Conference*, pages 508–519, 2022. URL https://proceedings.mlr.press/v168/zhi22a.html.

A Appendix

A.1 Metric

We evaluate trajectories by comparing them against their reference trajectory using the normalized dynamic time warping distance (DTWD) [20]. For two trajectories $x = \{x_i\}_{i\in\{1,\dots,t_1\}}$, $\bar{x} = \{\bar{x}_i\}_{i\in\{1,\dots,t_2\}}$ and some distance function $d(\cdot,\cdot)$, let DTWD be

$$\textbf{DTWD}(x,\bar{x}) = \frac{1}{t_1}\sum_{i=1}^{t_1}\left(\min_{\bar{x}_j\in\bar{x}} d(x_i,\bar{x}_j)\right) + \frac{1}{t_2}\sum_{i=1}^{t_2}\left(\min_{x_j\in x} d(\bar{x}_i,x_j)\right)$$

A.2 Comparisons to Other Models

In this section, we describe the models that we compare to, namely EFlow [32], SDD [30], and NCDS [6].

Euclideanizing Flow: Starting with EFlow, they parametrize their dynamical system by

$$\dot{x} = -G_\psi(x)^{-1}\nabla\Phi(\psi(x)), \tag{18}$$

where $\psi : \mathbb{R}^d \to \mathbb{R}^d$ is a diffeomorphism, $\Phi : \mathbb{R}^d \to \mathbb{R}$ is a convex potential function, and $G_\psi(x) = D\psi(x)^\top D\psi(x)$ is the induced Riemmanian metric. The dynamics (18) then has the interpretation of being the steepest descent for $\Phi\circ\psi$ on the Riemmanian manifold defined by metric G_ψ. Specifically, in [32], the convex potential function is defined to be $\Phi(y) = \|y - y^\star\|_2$, where $y^\star = \psi(x^\star)$ and the diffeomorphism is parametrized via a coupling layer [12]. Note that contraction properties of dynamics of these form were studied in [42]. Since Ψ is chosen to be convex but not strongly convex, these dynamics are only *weakly contracting*, i.e., satisfy (2) with $c = 0$.

Stable Deep Dynamics: In [30], the authors parametrize both an unconstrained vector field $\hat{f} : \mathbb{R}^d \to \mathbb{R}^d$ and a Lyapunov function $V : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ via neural networks. Then to enforce global exponential stability, at every point $x \in \mathbb{R}^d$, they project $\hat{f}(x)$ onto the convex set of points $\{u \in \mathbb{R}^d \mid \nabla V(x)^\top u \leq -\alpha V(x)\}$. They do this projection in closed-form so that the dynamics have the representation

$$\dot{x} = \hat{f}(x) - \nabla V(x)\frac{\mathrm{ReLU}(\nabla V(x)^\top \hat{f}(x) + \alpha V(x))}{\|\nabla V(x)\|_2^2}. \tag{19}$$

Under a smart parametrization of V using input-convex neural networks [3], the authors guarantee global exponential stability of (19) to the origin with rate α. Note that the dynamics (19) are continuous, but not necessarily differentiable. Due to this potential nonsmoothness, it is theoretically unknown whether these dynamics are contracting (since Theorem 2.8 in [16] requires at least some differentiability).

Neural Contractive Dynamical Systems: In [6], the authors enforce contractivity by directly enforcing negative definiteness of the Jacobian matrix of the vector field by parametrizing

$$Df(x) = -(J_\theta(x)^\top J_\theta(x) + \epsilon I_d), \tag{20}$$

where $J_\theta : \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is a neural network. Under this parametrization of the Jacobian, it is straightforward to see that (2) holds with $c = \epsilon$ and $M(x) = I_d$ for all x. To recover the vector field from the Jacobian, the fundamental theorem of calculus for line integrals (which can be seen as a version of the mean-value theorem) is utilized to express

$$\dot{x} = f(x) = f(x_0) + \int_0^1 Df((1-t)x_0 + tx)(x - x_0)dt, \tag{21}$$

where x_0 and $f(x_0)$ denote the initial condition and its velocity, respectively.

To express richer dynamics, NCDS also consists of an encoder and decoder for a lower-dimensional latent space. The dynamics in the latent space are constrained to be (21) and the encoder and decoder are parametrized using a VAE. Since, generally, the latent space is of lower dimension than the data space, the resulting dynamics are only guaranteed to be contracting on the submanifold defined by the image of the decoder.

A.3 Universal Representation

Our ELCD model equipped with a coupling layer-based transformation can represent any contracting dynamical system. It is known that any smooth nonlinear system with a hyperbolically stable fixed point can be exactly linearized inside its basin of attraction via a suitable diffeomorphism [25]. Effectively, our extended linearization parameterization is tasked with learning the stable linear part, while the coupling layers aim to learn and approximate this suitable diffeomorphism. As our ELCD parametrization can universally approximate any linear model, we simply need our coupling layer to universally approximate all diffeomorphisms. Indeed, [38] proved that the coupling layers are universal approximators for diffeomorphisms.

A.4 Additional Details on Datasets

In this section, we provide additional elaboration on the multi-link pendulum dataset and the Rosenbrock dataset.

Multi-link pendulum: The multi-link pendulum is a simple mechanical system which is the interconnection of n rigid links under the force of gravity. We specifically assume that there is damping on each of these links, which makes the resulting dynamical system stable and almost all trajectories converge to the downright equilibrium point. In this case, the state of the pendulum is described by the n pairs $(\theta_i, \dot{\theta}_i)$, where θ_i is the angle of the i-th link and $\dot{\theta}_i$ is its angular velocity. Thus, this dynamical system is $2n$ dimensional.

As was done in [30], we adapt the code from `http://jakevdp.github.io/blog/2017/03/08/triple-pendulum-chaos/` to generate data for n links.

Rosenbrock datset: The classical Rosenbrock function, see [34]

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2, \tag{22}$$

is a nonconvex function with global minimum at $(1, 1)$. Despite its apparent nonconvexity, it is known that a Riemannian gradient descent dynamics, is contracting with respect to a suitable Riemannian metric, see Example 3 in [42] for details. In optimization and in machine learning, the Rosenbrock function is a benchmark for the design of various optimizers and in the study of Riemannian optimization [19, 24].

For a higher-dimensional generalization of the Rosenbrock function, we consider

$$f(x) = \lambda_1(1 - x_1)^2 + \sum_{i=2}^{n} \lambda_i(x_i - x_{i-1}^2)^2, \tag{23}$$

where $\lambda_i > 0$ are parameters that affect the conditioning of the function. This generalization is still nonconvex and admits several saddle points. Note that the classic Rosenbrock corresponds to $n = 2, \lambda_1 = 1, \lambda_2 = 100$. The unique global minimum of this generalization is at $(1, 1, \dots, 1)$. The corresponding contracting Riemmanian gradient descent dynamics that find this global minimum are

$$\dot{x} = -G(x)^{-1}\nabla f(x), \tag{24}$$

where $G(x) = D\psi(x)^\top D\psi(x)$, where $\psi : \mathbb{R}^n \to \mathbb{R}^n$ is the mapping $\psi(x) = (\sqrt{\lambda_1}(1 - x_1), \sqrt{\lambda_2}(x_2 - x_1^2), \dots, \sqrt{\lambda_n}(x_n - x_{n-1}^2))$. Note that these dynamics were designed via a generalization of the procedure proposed in [42]. To generate data, we choose four initial points and evolved them according to (24).

A.5 Model Details

For the encoder ϕ, we use the composition of two coupling layers composed of rational-quadratic spline. The splines cover the range $\{-10, 10\}$ with 10 bins, and linearly extrapolated outside that range. The parameters of each spline are determined by residual networks, each containing two transform blocks with a hidden dimension of 30. A permutation and linear flow layer is placed before, in the middle, and after the two quadratic spline flow layers. For ELCD, we let the latent dimension size equal the data dimension size. P_a and P_s are implemented as two-layer neural networks with a hidden dimension of 16.

As the authors of NCDS have not yet made their code publicly available, we implemented NCDS to the best of our abilities. We used a latent dimension of size 2 for all datasets. This is in-line with the description in [6], and necessary given the extra cost from integrating the Jacobian in higher-dimensional latent spaces. We use the same encoder ϕ as for ELCD with an added un-padding layer as the last function which removes the last $d-2$ dimensions.

A.6 Training details

All experiments are trained with a batch size of 100, for 100 epochs, with an Adam optimizer and learning rate of 10^{-3}. All computation is done on CPUs.

NCDS [6] trained the model and the encoder to output x_{t+1} from x_t using a single Euler integration step: $x_{t+1} = \phi^{-1}(\phi(x_t) + dt * f(\phi(x_t))$. Their loss was the MSE between the predicted and true x_{t+1}. We empirically find better performance by training our model to directly predict the velocity vector $\dot{x}_{t+1} = \phi_J^{-1}(x_t) f(\phi(x_t))$, where $\phi_J^{-1}(x_t)$ is the Jacobian inverse of $\phi(x_t)$. Calculating the gradient of $\phi_J^{-1}(x_t)$ efficiently required a slight alteration to the original $\mathcal{M}$-flow [8] code, which we have included in our repository.

NeurIPS Paper Checklist

(i) **Claims**

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: We introduce a new model has global contraction gaurantess and is more expressible than prior work. We discuss the contraction properties and demonstrate our the expressibility experimentally.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

(ii) **Limitations**

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: See section 5

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

(iii) **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: Our main theorem in section (3.2) is supported by a complete proof.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

(iv) **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: Data details and model details are provided in the appendix. Code is also provided.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

(v) **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: Code to reproduce experiments is provided.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

(vi) **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: See Appendix A.6

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

(vii) **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: Standard Deviations are reported.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

(viii) **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: We explain in the appendix that we use CPUs.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

(ix) **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: We have reviewed and comply with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

(x) **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA] Replace by [Yes] , [No] , or [NA] .

Justification: There is no direct link between our work and a possible negative societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

(xi) **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA] Replace by [Yes] , [No] , or [NA] .

Justification: This paper poses no such risk.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

(xii) **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: Data and code from prior work are all cited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

(xiii) **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes] Replace by [Yes] , [No] , or [NA] .

Justification: Alongside the new ELCD model, we also provide the n-link pendulum dataset and Rosenbrock dataset in the supplementary material of the submission. Upon acceptance, we will release the datasets with a permissive license.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

(xiv) **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA] Replace by [Yes] , [No] , or [NA] .

Justification: The paper does not involve crowdsourcing.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

(xv) **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA] Replace by [Yes] , [No] , or [NA] .

Justification: The paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.