A dynamical systems view to Deep learning: contractivity and structure preservation.

Elena Celledoni

Department of Mathematical Sciences, NTNU, Trondheim, Norway

Exploiting Algebraic and Geometric Structure in
Time-Integration Methods,
Pisa, April 3-5, 2024

MSCA-SE: REMODEL Project ID: 101131557

Outline

- The dynamical systems view to deep learning, preservation of structure
- Adversarial attacks - robust NNs
- Contractivity of ODEs and of numerical integrators on Riemannian manifolds
- Optimisation on infinite dimensional Lie groups: invariance under reparametrization
- Learning ODEs from data.

Let $\mathcal{V}$ input space, $\mathcal{W}$ output space

$$\varphi : \mathcal{V} \to \mathcal{W}.$$

DNNs - **approximation theory**:

$$\varphi \approx \varphi_\theta$$

by a composition of L simpler maps (layers)

$$\varphi_\theta = \varphi_L \circ \varphi_{L-1} \circ \cdots \circ \varphi_1, \quad \varphi_\ell = \varphi_{\theta_\ell} \quad \varphi_{\theta_\ell} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell$$

$\mathcal{V}_0 = \mathcal{V}$ and $\mathcal{V}_L = \mathcal{W}$, each φ_ℓ depends on a finite number of parameters θ_ℓ.

Let $\mathcal{V}$ input space, $\mathcal{W}$ output space

$$\varphi : \mathcal{V} \to \mathcal{W}.$$

DNNs - **approximation theory**:

$$\varphi \approx \varphi_\theta$$

by a composition of L simpler maps (layers)

$$\varphi_\theta = \varphi_L \circ \varphi_{L-1} \circ \cdots \circ \varphi_1, \quad \varphi_\ell = \varphi_{\theta_\ell} \quad \varphi_{\theta_\ell} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell$$

$\mathcal{V}_0 = \mathcal{V}$ and $\mathcal{V}_L = \mathcal{W}$, each φ_ℓ depends on a finite number of parameters θ_ℓ.
Residual networks - **numerical ODEs**: $\mathcal{V}_{\ell-1} = \mathcal{V}_\ell = \mathcal{V}$ a compact subdomain of $\mathbb{R}^N$ and

$$\varphi_{\theta_\ell} = \mathrm{id} + hX_{\theta_\ell}, \quad X_{\theta_\ell} : x \mapsto \sigma(A_\ell x + b_\ell), \quad \theta_\ell := (A_\ell, b_\ell)$$

can be seen as the forward Euler discretization of the flow map of the ODE

$$\dot{y} = \sigma(A(t)y(t) + b(t)), \qquad y(0) = x, \qquad t \in [0, h]$$

(Haber and Ruthotto 2017, and E 2017).

Let $\mathcal{V}$ input space, $\mathcal{W}$ output space

$$\varphi : \mathcal{V} \to \mathcal{W}.$$

DNNs - **approximation theory**:

$$\varphi \approx \varphi_\theta$$

by a composition of L simpler maps (layers)

$$\varphi_\theta = \varphi_L \circ \varphi_{L-1} \circ \cdots \circ \varphi_1, \quad \varphi_\ell = \varphi_{\theta_\ell} \quad \varphi_{\theta_\ell} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell$$

$\mathcal{V}_0 = \mathcal{V}$ and $\mathcal{V}_L = \mathcal{W}$, each φ_ℓ depends on a finite number of parameters θ_ℓ.
Residual networks - **numerical ODEs**: $\mathcal{V}_{\ell-1} = \mathcal{V}_\ell = \mathcal{V}$ a compact subdomain of $\mathbb{R}^N$ and

$$\varphi_{\theta_\ell} = \mathrm{id} + hX_{\theta_\ell}, \quad X_{\theta_\ell} : x \mapsto \sigma(A_\ell x + b_\ell), \quad \theta_\ell := (A_\ell, b_\ell)$$

can be seen as the forward Euler discretization of the flow map of the ODE

$$\dot{y} = \sigma(A(t)y(t) + b(t)), \qquad y(0) = x, \qquad t \in [0, h]$$

(Haber and Ruthotto 2017, and E 2017).
Learning - **variational methods**: optimising a cost function (distance) with respect to all the parameters

$$\min_{\varphi = \varphi_{\theta_L} \circ \cdots \circ \varphi_{\theta_1}} E(\varphi) = \min_{\{\theta_\ell\}_{\ell=1}^L} E(\varphi_{\theta_L} \circ \cdots \circ \varphi_{\theta_1})$$

is the **discretization of the optimal control problem**:

$$\inf_{A(t), b(t)} E(y(T)), \quad \text{subject to} \quad \dot{y} = \sigma(Ay + b), \quad y(0) = x, \quad t \in [0, T].$$

Transitions in Runge–Kutta methods – Data Spiral

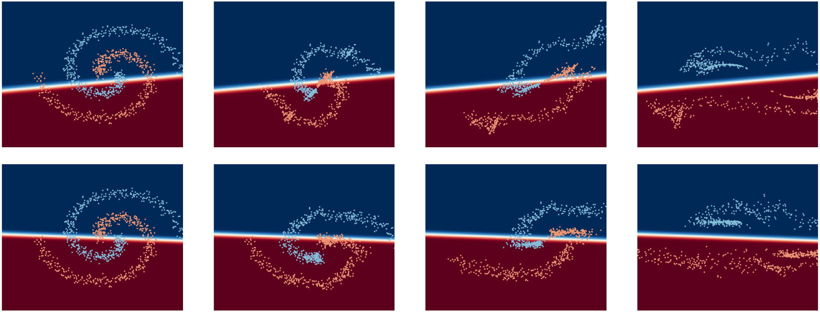


Figure: Snap shots of the transition from initial to final state through the network with the *Spiral* data set. Top, ResNet/Euler, and bottom, Runge-Kutta(4).

Benning, EC, Ehrhardt, Owren, Schönlieb, *Deep learning as optimal control problems: models and numerical methods*, JCD, 2019

Let $\mathcal{V}$ input space, $\mathcal{W}$ output space. Neural networks inherit structure that is preserved under composition

$$\varphi \approx \varphi_\theta = \varphi_{\theta_L} \circ \varphi_{\theta_{L-1}} \circ \cdots \circ \varphi_{\theta_1}, \qquad \varphi_{\theta_\ell} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell, \quad \mathcal{V}_0 = \mathcal{V},\ \mathcal{V}_L = \mathcal{W},$$

by imposing structure on the layers φ_{θ_ℓ} (e.g. volume preservation), φ_θ inherits the same structural properties.

Let $\mathcal{V}$ input space, $\mathcal{W}$ output space. Neural networks inherit structure that is preserved under composition

$$\varphi \approx \varphi_\theta = \varphi_{\theta_L} \circ \varphi_{\theta_{L-1}} \circ \cdots \circ \varphi_{\theta_1}, \qquad \varphi_{\theta_\ell} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell, \quad \mathcal{V}_0 = \mathcal{V},\ \mathcal{V}_L = \mathcal{W},$$

by imposing structure on the layers φ_{θ_ℓ} (e.g. volume preservation), φ_θ inherits the same structural properties.

Dynamical systems can be classified according to the **group of diffeomorphisms** $(\mathcal{G}, \circ)$ to which they belong.

Let $\mathcal{V}$ input space, $\mathcal{W}$ output space. Neural networks inherit structure that is preserved under composition

$$\varphi \approx \varphi_\theta = \varphi_{\theta_L} \circ \varphi_{\theta_{L-1}} \circ \cdots \circ \varphi_{\theta_1}, \qquad \varphi_{\theta_\ell} : \mathcal{V}_{\ell-1} \to \mathcal{V}_\ell, \quad \mathcal{V}_0 = \mathcal{V},\ \mathcal{V}_L = \mathcal{W},$$

by imposing structure on the layers φ_{θ_ℓ} (e.g. volume preservation), φ_θ inherits the same structural properties.

Dynamical systems can be classified according to the **group of diffeomorphisms** $(\mathcal{G}, \circ)$ to which they belong.

Learning restricted to a **finite dimensional space** $\mathcal{G}_\theta$ contained in $\mathcal{G}$

$$\mathcal{G}_\theta := \{\varphi_\theta \in \mathcal{G} \,|\, \varphi_\theta = \varphi_{\theta_L} \circ \cdots \circ \varphi_{\theta_1}\}, \quad L \quad \text{fixed}$$

and

$$\min_{\varphi_\theta \in \mathcal{G}_\theta} E(\varphi) = \min_{\{\theta_\ell\}_{\ell=1}^L} E(\varphi_{\theta_L} \circ \cdots \circ \varphi_{\theta_1}).$$

Strategy for constructing NNs: choose vector fields in the correct Lie algebra use exact flows or numerical methods preserving the structure to obtain layers φ_{θ_ℓ} in the correct group.

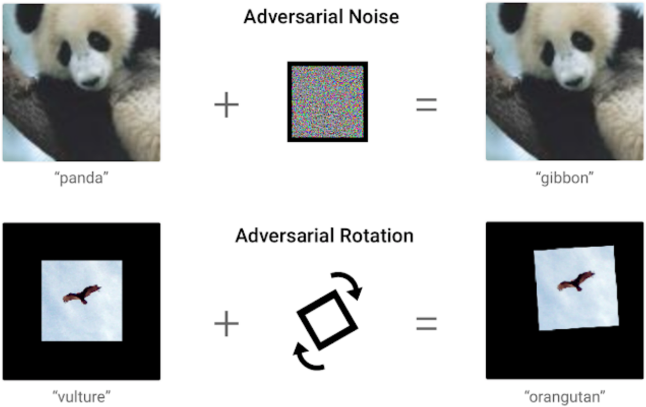
Strategy for constructing NNs: choose vector fields in the correct Lie algebra use exact flows or numerical methods preserving the structure to obtain layers φ_{θ_ℓ} in the correct group.

- We can design: 1-Lipschitz networks, invertible networks, volume preserving networks, symplectic networks, mass preserving networks.
- We can compose vector fields in different classes to enhance expressivity (prove universal approximation results).
- We can construct 1-Lipschitz networks which are expressive and robust against adversarial attacks.

Advantage: it can be easier to impose structure on the vector fields than on the corresponding flows.

EC, Murari, Owren, Schönlieb and Sherry, *Dynamical systems based neural networks*, 2023, SISC

(In)stability – adversarial attacks



https://ai.googleblog.com/2018/09/

Residual networks:

$$\varphi \approx \varphi_\theta = \varphi_{\theta_L} \circ \varphi_{\theta_{L-1}} \circ \cdots \circ \varphi_{\theta_1}, \qquad \varphi_{\theta_\ell} : \mathcal{V} \to \mathcal{V},$$

$\mathcal{V}$ a compact subdomain of $\mathbb{R}^N$ and

$$\varphi_{\theta_\ell} = \mathrm{id} + hX_{\theta_\ell}, \quad X_{\theta_\ell} : x \mapsto B_\ell \sigma(A_\ell x + b_\ell), \quad \theta_\ell := (B_\ell, A_\ell, b_\ell)$$

forward Euler numerical integration of the ODE

$$\dot{y} = B(t)\sigma(A(t)y(t) + b(t)), \qquad y(0) = x, \qquad t \in [0, h].$$

Residual networks:

$$\varphi \approx \varphi_\theta = \varphi_{\theta_L} \circ \varphi_{\theta_{L-1}} \circ \cdots \circ \varphi_{\theta_1}, \qquad \varphi_{\theta_\ell} : \mathcal{V} \to \mathcal{V},$$

$\mathcal{V}$ a compact subdomain of $\mathbb{R}^N$ and

$$\varphi_{\theta_\ell} = \mathrm{id} + hX_{\theta_\ell}, \quad X_{\theta_\ell} : x \mapsto B_\ell \sigma(A_\ell x + b_\ell), \quad \theta_\ell := (B_\ell, A_\ell, b_\ell)$$

forward Euler numerical integration of the ODE

$$\dot{y} = B(t)\sigma(A(t)y(t) + b(t)), \qquad y(0) = x, \qquad t \in [0, h].$$

- We want to be able to guarantee that the layer φ_ℓ is a contractive map (when necessary), i.e.

  $$\|\varphi_\ell(y_2) - \varphi_\ell(y_1)\| < \|y_2 - y_1\|,$$

  so that we can compose contractive and non-contractive layers to construct a neural network with Lipschitz constant equal to 1.
- Then we can use known theory of numerical stability of contractive ODEs.

Contractivity of the underlying ODE

A vector field $X(t,y)$ is **contractive** in L^2-norm if there is $\nu < 0$ such that for all y_1, y_2 and $t \in [0,T]$:

$$\langle X(t,y_2) - X(t,y_1), y_2 - y_1 \rangle \leq \nu \|y_2 - y_1\|^2.$$

This implies that for any two integral curves $y(t)$ and $z(t)$

$$\|y(t) - z(t)\| \leq e^{t\nu} \|y(0) - z(0)\|.$$

A vector field $X(t, y)$ is **contractive** in L^2-norm if there is $\nu < 0$ such that for all y_1, y_2 and $t \in [0, T]$:

$$\langle X(t, y_2) - X(t, y_1), y_2 - y_1 \rangle \leq \nu \|y_2 - y_1\|^2.$$

This implies that for any two integral curves $y(t)$ and $z(t)$

$$\|y(t) - z(t)\| \leq e^{t\nu} \|y(0) - z(0)\|.$$

The vector field

$$X(t, y(t)) = -A(t)^T \sigma(A(t)y(t) + b(t)),$$

with σ increasing function, $A \in \mathbb{R}^{n \times k}$, $b \in \mathbb{R}^n$, **is contractive**.

EC, Ehrhardt, Etmann, McLachlan, Owren, Schönlieb, Sherry, *Structure preserving deep learning*, EJAM, 2021.

Contractivity of explicit Runge-Kutta methods: forward Euler

Theorem (Dahlquist and Jeltsch, 1979)
Suppose X satisfies the **monotonicity condition**

$$\langle X(t,y_2) - X(t,y_1), y_2 - y_1 \rangle \leq \bar{\nu} \| X(t,y_2) - X(t,y_1) \|^2, \quad \bar{\nu} < 0.$$

Then, if the stepsize h satisfies

$$h \leq -2\bar{\nu},$$

the **forward Euler method is contractive**.

Theorem (Dahlquist and Jeltsch, 1979)
Suppose X satisfies the **monotonicity condition**

$$\langle X(t,y_2) - X(t,y_1), y_2 - y_1 \rangle \leq \bar{\nu} \| X(t,y_2) - X(t,y_1) \|^2, \quad \bar{\nu} < 0.$$

Then, if the stepsize h satisfies

$$h \leq -2\bar{\nu},$$

the **forward Euler method is contractive**.

Proposition
For σ non decreasing and L-Lipschitz, the vector field

$$X(t,y) = -A(t)^T \sigma(A(t)y + b),$$

with $A \in \mathbb{R}^{n\times k}$, $b \in \mathbb{R}^n$, satisfies the monotonicity condition with $\bar{\nu} = -\frac{1}{\|A\|^2 L}$.

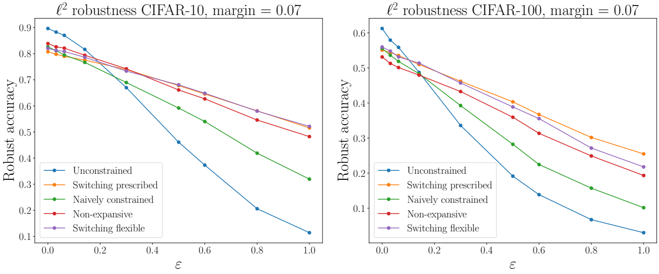
Remark X is a gradient vector field:
$\dot{y} = -\nabla_y V, \qquad V(t, y(t)) = \langle \gamma\big(A(t)y(t) + b(t)\big), \mathbb{1} \rangle, \quad \gamma' = \sigma.$

Sherry, EC, Ehrhardt, Murari, Owren and Schönlieb, *Designing Stable Neural Networks using Convex Analysis and ODEs*, 2024, Physica D, to appear.

Robust classification of CIFAR10 and CIFAR100

$$\varphi_\ell(x) = x - h_1\, P^T \sigma(Px + p) \quad \text{contractive}$$
$$\psi_\ell(x) = x + h_2\, Q^T \sigma(Qx + q) \quad \text{expansive}$$
$$\sigma(x) = \max\left\{x, \frac{x}{2}\right\},\; P^T P = I,\; Q^T Q = I.$$

Using orthogonal convolutional NNs. by Wang et al., 2020. Adversarial examples using Foolbox.



EC, Murari, Owren, Schönlieb and Sherry, *Dynamical systems based neural networks*, 2023, SISC

- $(\mathcal{M}, g)$ a Riemannian manifold, $g(u, v) = \langle u, v \rangle$
- ∇ is the Levi-Civita connection induced by g
- X and Y vector fields on $\mathcal{M}$: $\nabla_X Y$ denotes the covariant derivative on $\mathcal{M}$
- $\gamma(t) = exp_p(t\, v_p)$ Riemannian exponential
- $d(p, q) = \inf_{\gamma_{p\to q}} \ell\,(\gamma_{p\to q}), \quad \ell(\gamma) = \int_a^b \|\dot{\gamma}(t)\| dt$

- $(\mathcal{M}, g)$ a Riemannian manifold, $g(u,v) = \langle u, v\rangle$
- ∇ is the Levi-Civita connection induced by g
- X and Y vector fields on $\mathcal{M}$: $\nabla_X Y$ denotes the covariant derivative on $\mathcal{M}$
- $\gamma(t) = exp_p(t\, v_p)$ Riemannian exponential
- $d(p,q) = \inf_{\gamma_{p\to q}} \ell(\gamma_{p\to q}), \quad \ell(\gamma) = \int_a^b \|\dot{\gamma}(t)\| dt$

Monotonicity condition: for $\mathcal{U} \subset \mathcal{M}$ a vector field X satisfies the monotonicity condition on $\mathcal{U}$ iff $\forall x \in \mathcal{U}$ $v_x \in T_x\mathcal{M}$ then

$$\langle \nabla_{v_x} X, v_x\rangle \le \nu \|v_x\|^2.$$

- $(\mathcal{M}, g)$ a Riemannian manifold, $g(u, v) = \langle u, v\rangle$
- ∇ is the Levi-Civita connection induced by g
- X and Y vector fields on $\mathcal{M}$: $\nabla_X Y$ denotes the covariant derivative on $\mathcal{M}$
- $\gamma(t) = exp_p(t\, v_p)$ Riemannian exponential
- $d(p, q) = \inf_{\gamma_{p\to q}} \ell(\gamma_{p\to q}), \quad \ell(\gamma) = \int_a^b \|\dot{\gamma}(t)\| dt$

Monotonicity condition: for $\mathcal{U} \subset \mathcal{M}$ a vector field X satisfies the monotonicity condition on $\mathcal{U}$ iff $\forall x \in \mathcal{U}\ v_x \in T_x\mathcal{M}$ then

$$\langle \nabla_{v_x} X, v_x\rangle \le \nu \|v_x\|^2.$$

Then one can prove that for $\mathcal{U}$ geodesically convex, with $y(t) = \exp(tX)y_0$, $z(t) = \exp(tX)z_0$ in $\mathcal{U}$ $\forall t \in [0, T]$ it holds

$$d(y(t), z(t)) \le e^{t\nu} d(y_0, z_0), \quad \forall t \in [0, T].$$

Contractivity of numerical integrators on Riemannian manifolds

- $(\mathcal{M}, g)$ a Riemannian manifold, $g(u,v) = \langle u, v\rangle$
- ∇ is the Levi-Civita connection induced by g
- X and Y vector fields on $\mathcal{M}$: $\nabla_X Y$ denotes the covariant derivative on $\mathcal{M}$
- $\gamma(t) = exp_p(t\, v_p)$ Riemannian exponential
- $d(p,q) = \inf_{\gamma_{p\to q}} \ell(\gamma_{p\to q}), \quad \ell(\gamma) = \int_a^b \|\dot\gamma(t)\| dt$

Monotonicity condition: for $\mathcal{U} \subset \mathcal{M}$ a vector field X satisfies the monotonicity condition on $\mathcal{U}$ iff $\forall x \in \mathcal{U}\ \ v_x \in T_x\mathcal{M}$ then

$$\langle \nabla_{v_x} X, v_x\rangle \leq \nu \|v_x\|^2.$$

Then one can prove that for $\mathcal{U}$ geodesically convex, with $y(t) = \exp(tX)y_0$, $z(t) = \exp(tX)z_0$ in $\mathcal{U}$ $\forall t \in [0,T]$ it holds

$$d(y(t), z(t)) \leq e^{t\nu} d(y_0, z_0), \quad \forall t \in [0,T].$$

Non-expansiveness when X is forward complete, $\mathcal{U}$ is forward X-invariant and $\nu \leq 0$.

- M. Kunzinger and H. Schichl and R. Steinbauer and J. A. Vickers, 2006, Revista Matemática Complutense
- J. W. Simpson-Porco and F. Bullo, Contraction theory on Riemannian manifolds, Systems & Control Letters, 2014

Definition: Suppose

- X is contractive on $\mathcal{U} \subset \mathcal{M}$,
- $\phi_{h,X} : \mathcal{M} \to \mathcal{M}$ is a numerical method approximating $\exp(tX)p$ and is well defined for all $h \geq 0$,
- $\mathcal{U}$ is forward $\phi_{h,X}$-invariant for all $h \geq 0$ and forward X-invariant

then the method is said to be **B-stable** iff

$$d(\phi_{h,X}(y_0), \phi_{h,X}(z_0)) \leq d(y_0, z_0), \quad \forall h \geq 0.$$

Definition: Suppose

- X is contractive on $\mathcal{U} \subset \mathcal{M}$,
- $\phi_{h,X} : \mathcal{M} \to \mathcal{M}$ is a numerical method approximating $\exp(tX)p$ and is well defined for all $h \geq 0$,
- $\mathcal{U}$ is forward $\phi_{h,X}$-invariant for all $h \geq 0$ and forward X-invariant

then the method is said to be **B-stable** iff

$$d(\phi_{h,X}(y_0), \phi_{h,X}(z_0)) \leq d(y_0, z_0), \quad \forall h \geq 0.$$

Geodesic Implicit Euler

$$y_n = \exp_{y_{n+1}}(-hX(y_{n+1})).$$

B-stability of integrators on Riemannian manifolds

Definition: Suppose

- X is contractive on $\mathcal{U} \subset \mathcal{M}$,
- $\phi_{h,X} : \mathcal{M} \to \mathcal{M}$ is a numerical method approximating $\exp(tX)p$ and is well defined for all $h \geq 0$,
- $\mathcal{U}$ is forward $\phi_{h,X}$-invariant for all $h \geq 0$ and forward X-invariant

then the method is said to be **B-stable** iff

$$d(\phi_{h,X}(y_0), \phi_{h,X}(z_0)) \leq d(y_0, z_0), \quad \forall\, h \geq 0.$$

Geodesic Implicit Euler

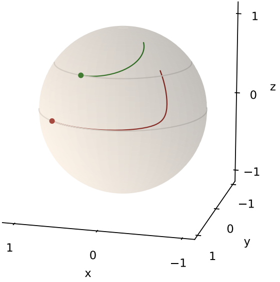
$$y_n = \exp_{y_{n+1}}(-hX(y_{n+1})).$$

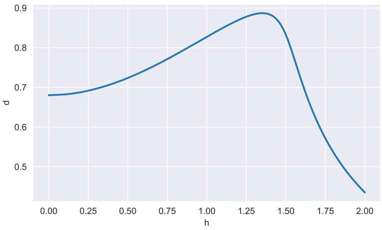
Theorem If $\mathcal{M}$ is a Riemannian manifold with non-positive sectional curvature then the geodesic implicit Euler method is B-stable.

Example Space of $n \times n$ symmetric positive definite matrices.

Arnold, EC, Cokaj, Owren, Tumiotto, *Contractivity of numerical integrators on Riemannian manifolds*, 2024, JCD.

The sphere has positive sectional curvature equal to 1:





- Non-expansive vector field (on the northern hemisphere)

$$\dot{y} = X(y) = a \times y, \qquad a = [0, 0, 1].$$

- (Left) One step of GIE applied with increasing step size h, starting from two different initial values.
- (Right) Geodesic distance: $d(y_1, z_1)$ plotted as a function of h, where $y_0 = \exp_{y_1}(-hX(y_1)), \quad z_0 = \exp_{z_1}(-hX(z_1))$.

Neural networks for regularising inverse problems

- **Variational regularization in image processing**
  clean images $\hat{u}$ are recovered from measurements y by minimising a trade-off between
  1. $E_y(u) := d(A(u), y)$ the **data fit** and
  2. $R(u)$ **penalty function** encoding prior knowledge

$$\hat{u} = \arg\min_u E_y(u) + R(u).$$

- **Splitting methods for optimisation**: split the objective function in two or more terms, each easier to optimise.
- **Proximal gradient** is a sort of gradient descent where the gradient flow is approximated by an implicit-explicit time-stepping. The implicit part corresponds to the **proximal operator**:

Proposition:

$$\mathrm{prox}_{hR}\, u = \arg\min_{u'} \|u - u'\|_2 + h\, R(u').$$

To solve the optimization problem

$$\hat{u} = \arg\min_u E_y(u) + R(u)$$

we use

Proximal gradient descent

Input: measurements y, initial estimate u_0

for $\ell = 1, \dots, N$ **do**

$$u^{[\ell+1]} = \mathrm{prox}_{hR}(u^{[\ell]} - h\nabla E_y(u^{[\ell]}))$$

end for

Plug-and-Play: replace prox_{hR} with a (non-expansive) neural network $\widehat{\mathrm{prox}}_{h,\ell}$, learning the de-noiser form data.

Definition An operator $\mathcal{A} : \mathbb{R}^d \to \mathbb{R}^d$ is α-averaged if $\exists$ a non expansive operator $T : \mathbb{R}^d \to \mathbb{R}^d$ s.t.

$$\mathcal{A} = \alpha T + (1 - \alpha) I_d, \quad \alpha \in (0, 1).$$

Definition An operator $\mathcal{A} : \mathbb{R}^d \to \mathbb{R}^d$ is α-averaged if $\exists$ a non expansive operator $T : \mathbb{R}^d \to \mathbb{R}^d$ s.t.

$$\mathcal{A} = \alpha T + (1-\alpha) I_d, \quad \alpha \in (0,1).$$

Theorem (Hertrich, Neumayer, Steidl)
Let $E : \mathbb{R}^m \to \mathbb{R}$ be convex and differentiable with L-Lipschitz continuous gradient and let $\widehat{\mathrm{prox}}_{h,\ell} : \mathbb{R}^m \to \mathbb{R}^m$ be averaged. Then, for any $0 < h < \frac{2}{L}$, the sequence $\{u^{[\ell]}\}_\ell$ generated by

Proximal gradient descent-PnP:

for $\ell = 1, \dots, N$ **do**

$$u^{[\ell+1]} = \widehat{\mathrm{prox}}_{h,\ell}(u^{[\ell]} - h \nabla E_y(u^{[\ell]}))$$

end for

converges.

J Hertrich, S Neumayer, G Steidl, *Convolutional Proximal Neural Networks and Plug-and-Play Algorithms* , Lin. Alg. and Appl.

PnP with ResNet and non-expansive networks

- Using $f(t, y) = -A^T\sigma(Ay + b)$ we can construct residual neural networks that are provably non-expansive (1-Lipschitz) and averaged.
- **J Hertrich, S Neumayer, G Steidl.** Averagedness together with $E_y(u)$ convex, differentiable and ∇E_y is L-Lipschitz, is sufficient to prove convergence of PnP algorithms.

- Using $f(t,y) = -A^T\sigma(Ay+b)$ we can construct residual neural networks that are provably non-expansive (1-Lipschitz) and averaged.
- **J Hertrich, S Neumayer, G Steidl.** Averagedness together with $E_y(u)$ convex, differentiable and ∇E_y is L-Lipschitz, is sufficient to prove convergence of PnP algorithms.

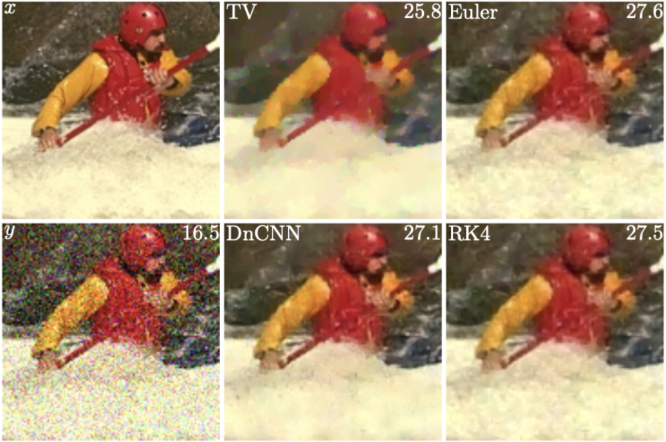
Theorem (Sherry)
Let For σ non decreasing and L-Lipschitz, the vector field $A \in \mathbb{R}^{n\times k}$, $b \in \mathbb{R}^n \sigma, A, b$ be as in and let $\alpha \in (0,1)$. A single layer of the proposed architecture, $\varphi(x) = x - hA^\top\sigma(Ax+b)$, is α-averaged if

$$h\|A\|^2 \leq 2\alpha/L. \tag{1}$$

Remark Composition of m operators $\mathcal{A}_i$, $i = 1, \ldots, m$ which are α_i averaged is α averaged for a certain α.

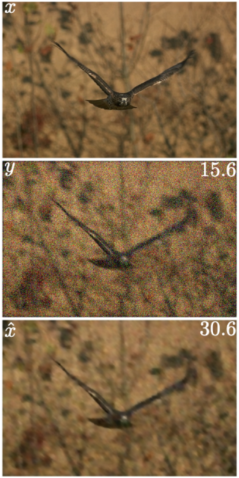
Sherry, EC, Ehrhardt, Murari, Owren and Schönlieb, *Designing Stable Neural Networks using Convex Analysis and ODEs*, 2024, Physica D: Nonlinear Phenomena.

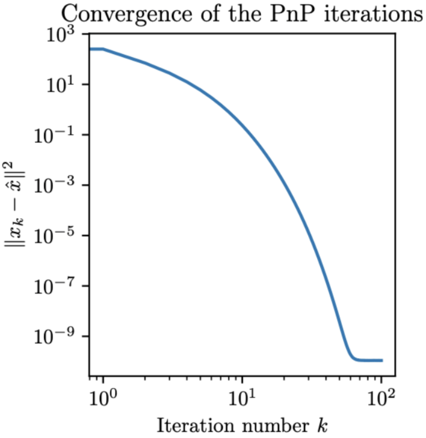
Denoising with PnP (Courtesy of F. Sherry)



Sherry, EC, Ehrhardt, Murari, Owren and Schönlieb, *Designing Stable Neural Networks using Convex Analysis and ODEs*, 2024, Physica D.

Convergence with PnP (Courtesy of F. Sherry)



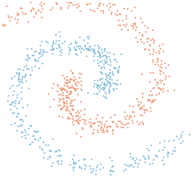


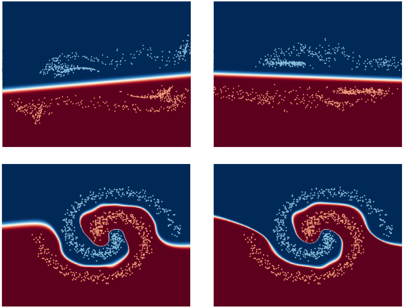
Sherry, EC, Ehrhardt, Murari, Owren and Schönlieb, *Designing Stable Neural Networks using Convex Analysis and ODEs*, 2024, Physica D.

Learning optimal parametrizations for shapes

Binary classification, connections to shape analysis

Example: *Spiral* **– Binary classification – Training and generalization**





$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \|\varphi_\theta(\mathrm{x}_i) - \mathrm{c}_i\|_2$$

Figure: Transformed points (top), prediction (bottom) for ResNet, 15 layers.

- Transformation of the domain via a differential equation (a diffeomorphism) before comparison, is relevant also in shape analysis.
- We will see how optimisation on the diffeomorphims group occurs in shape analysis.

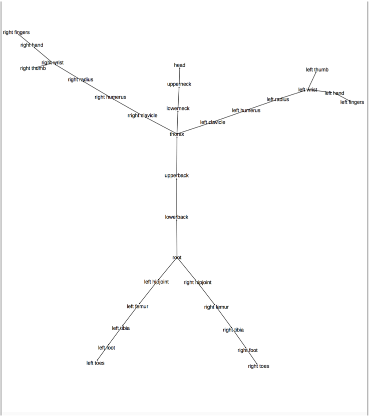
- Shape analysis is a framework for treating complex data and obtain metrics on the data spaces.
  Examples are spaces of *curves*, *time-signals*, *surfaces*, *images*, *probability distributions.*
- Shape analysis can be used for *data classification* or for *data generation* (interpolation) or *prediction* (extrapolation).

In this talk

Shapes are *unparametrized curves or surfaces* taking values in a vector space or on a manifold.

Skeletal animation

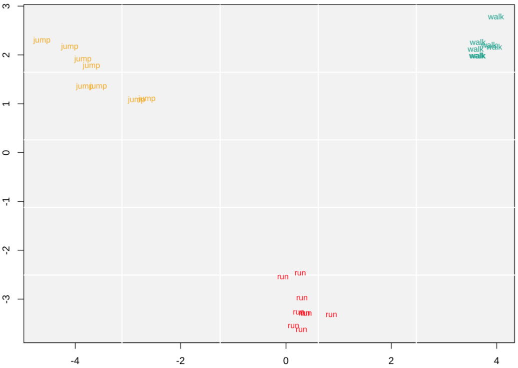
Skeleton consisting of bones connected by joints. One rotation for each joint.



Human activity: $\alpha : [0, T] \rightarrow \mathcal{J}$, $\mathcal{J} = SO(3)^n$, $[0, T]$ interval of time.

- Data obtained by motion capturing.
- Motion manipulation is the processing of the data.

Classifying running, walking, jumping animations as shapes



- E. C., P. E. Lystad and N. Tapia, GSI Proceedings, 2019,
- E. C., M. Eslitzbichler and A. Schmeding, JGM, 2016

Structure preserving shape analysis

- Geometry of rotations is preserved.
- Reparametration invariance of the distance between time-curves.
- Costly optimization algorithms, e.g. dynamic programming, to find the optimal reparametrization.

Definition of **shapes** via an equivalence relation: let $\mathrm{I} \subset \mathbb{R}$ an interval, consider

$$\mathcal{P} := \mathrm{Imm}(\mathrm{I}, \mathbb{R}^n) = \{c \in C^\infty(\mathrm{I}, \mathbb{R}^n) \mid \dot{c}(t) \neq 0\},$$

$\mathcal{P}$ is called pre-shape space and is an **infinite dimensional manifold**. Let $c_0, c_1 \in \mathcal{P}$ then

$$c_0 \sim c_1 \Longleftrightarrow \exists\, \varphi : \quad c_0 = c_1 \circ \varphi$$

with $\varphi \in \mathrm{Diff}^+(\mathrm{I})$ a orientation preserving diffeomorphism on I

Shape space:

$$\mathcal{S} := \mathrm{Imm}(\mathrm{I}, \mathbb{R}^n)/\mathrm{Diff}^+(\mathrm{I})$$

Applications often require a **distance** function to measure similarities between shapes. Let $d_{\mathcal{P}}$ be a distance function on $\mathcal{P}$

Distance on $\mathcal{S}$:

$$d_{\mathcal{S}}([c_0],[c_1]) := \inf_{\varphi \in \mathrm{Diff}^+(\mathrm{I})} d_{\mathcal{P}}(c_0, c_1 \circ \varphi). \tag{2}$$

Condition guaranteeing that $d_{\mathcal{S}}$ is well defined:

If $d_{\mathcal{P}}$ is such that

$$d_{\mathcal{P}}(c_0, c_1) = d_{\mathcal{P}}(c_0 \circ \varphi, c_1 \circ \varphi) \quad \forall \varphi \in \mathrm{Diff}^+(\mathrm{I}), \tag{3}$$

then $d_{\mathcal{S}}([c_0],[c_1])$ is well defined.

Distance on $\mathcal{P}$ via **SRVT** *and Q-transform*

One can proceed first transforming the curve transform $c \mapsto q = Q(c)$ and then computing L_2 distances: let $\mathcal{P} = \mathrm{Imm}(I, \mathbb{R}^n)$,

$$Q : \mathcal{P} \to C^\infty(I, \mathbb{R}^n), \quad c \mapsto q, \quad Q(c) := \begin{cases} \frac{\dot{c}}{\sqrt{\|\dot{c}\|}} & \text{SRVT} \\ \sqrt{\|\dot{c}(\cdot)\|}\, c(\cdot) & Q-\text{transform} \end{cases}$$

on $C^\infty(I, \mathbb{R}^n)$ we will use the L_2 metric:

$$d_{\mathcal{P}}(c_0, c_1) = d_{L_2}(Q(c_0), Q(c_1)) = \|q_0 - q_1\|_{L_2}.$$

- Both SRVT and Q-transform are **equivariant** with respect to reparametrisations, i.e.

$$Q(c \circ \gamma)(t) = \sqrt{\dot{\gamma}(t)} \cdot (Q(c) \circ \gamma)(t),$$

as a consequence $d_{\mathcal{P}}$ is reparametrization invariant.

- A. Srivastava, E. Klassen, S.H. Joshi, and I.H. Jermyn. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011.
- M. Mani, S. Kurtek, C. Barillot, A. Srivastava, IEEE Symposium on Biomedical Imaging, 2010.

Reformulation of the optimal reparametrization problem

Optimal reparametrization problem: given curves c_0 and c_1 with $q_0 := Q(c_0)$, and $q_1 := Q(c_1)$,

$$\inf_{\varphi \in \mathrm{Diff}^+(\mathrm{I})} E(\varphi), \qquad E(\varphi) := \| q_0 - \sqrt{\dot{\varphi}} \cdot (q_1 \circ \varphi) \|_{L_2}^2,$$

$$d_{\mathcal{S}}([c_0],[c_1]) = \inf_{\varphi \in \mathrm{Diff}^+(\mathrm{I})} E(\varphi).$$

Optimal reparametrization problem: given curves c_0 and c_1 with $q_0 := Q(c_0)$, and $q_1 := Q(c_1)$,

$$\inf_{\varphi \in \mathrm{Diff}^+(\mathrm{I})} E(\varphi), \qquad E(\varphi) := \| q_0 - \sqrt{\dot{\varphi}} \cdot (q_1 \circ \varphi) \|^2_{L_2},$$

$$d_{\mathcal{S}}([c_0],[c_1]) = \inf_{\varphi \in \mathrm{Diff}^+(\mathrm{I})} E(\varphi).$$

Optimisation problem on an infinite dimensional Lie group $\mathrm{Diff}^+(\mathrm{I})$, with "Lie algebra" $T_{\mathrm{id}}\mathrm{Diff}^+(\mathrm{I})$, $I = [0,1]$. We parametrize the diffeomorphisms with deep neural networks as follows:

Consider a basis $v_1, v_2, \dots$ of $T_{\mathrm{id}}\mathrm{Diff}^+(\mathrm{I})$ write

$$\varphi_\theta = (\mathrm{id} + h_1 X_{\theta_1}) \circ \cdots \circ (\mathrm{id} + h_L X_{\theta_L}).$$

$$X_{\theta_\ell} = \sum_{j=1}^{M} \beta_j^\ell v_j, \qquad \theta_\ell = \{\beta_j^\ell\}_{j=1,..,M}^{\ell=1,\dots,L}$$

Optimise on these "approximate diffeomorphisms".

We can show that finite compositions of diffeomorphisms of the type

$$\varphi_{\theta_\ell} = \mathrm{id} + X_{\theta_\ell}, \qquad \ell = 1, \ldots, L$$

with X_{θ_ℓ} a 1-Lipschitz vector field can be used to describe the whole group of

- diffeomorphisms fixing the boundary of a compact set $\Omega \subseteq \mathbb{R}^d$;
- diffeomorphisms on a cube $\Omega = [0, 1]^d$.

Expressivity: diffeomorphisms fixing the boundary

- $\Omega \in \mathbb{R}^d$ compact, connected subset with dense interior.
- $\mathrm{Diff}_\partial(\Omega)$ diffeomerphisms fixing the boundary.
- $T_{\mathrm{id}}\mathrm{Diff}_\partial(\Omega) = C^\infty_\partial(\Omega, \mathbb{R}^d)$ Lie algebra.

Global chart given by

$$\kappa \colon \mathrm{Diff}_\partial(\Omega) \to C^\infty_\partial(\Omega, \mathbb{R}^d), \quad \phi \mapsto \phi - \mathrm{id}_\Omega. \tag{4}$$

For vector fields $f \in \kappa(\mathrm{Diff}_\partial(\Omega))$, the inverse $\kappa^{-1}(f) = f + \mathrm{id}_\Omega \in \mathrm{Diff}_\partial(\Omega)$ and we can generate all elements in $\mathrm{Diff}_\partial(\Omega)$ this way.
Every diffeomorphism fixing the boundary can be expressed as a vector field (vanishing on the boundary) plus the identity.
However there are practical problems:

1. The image of the global chart $\kappa(\mathrm{Diff}_\partial(\Omega))$ is difficult to describe.
2. What can we approximate by restricting to a finite-dimensional subspace of the Lie algebra.
3. Not all diffeomorphisms of interest fix the boundary.

Example: 1D Let $\Omega = [a,b]$. Then a vector field f in $C^\infty_\partial([a,b],\mathrm{R})$ is in the image of κ if $f'(x) > -1$ for all $x \in [a,b]$ since then $\mathrm{id}_{[a,b]} + f$ will be monotonically increasing.

More complicated to describe $\kappa(\mathrm{Diff}_\partial(\Omega))$ in several dimensions, we use 1-Lipschitz vector fields:

$$\mathcal{U}_1 := \{f \in C^\infty_\partial(\Omega,\mathrm{R}^d) : \mathrm{Lip}(f) < 1\}$$

is an open 0-neighbourhood in $T_{\mathrm{id}}\mathrm{Diff}_\partial(\Omega)$.
A sufficient criterion:

Lemma

The map $\kappa^{-1}(f) = \mathrm{id}_\Omega + f$ is an element of $\mathrm{Diff}_\partial(\Omega)$ for all $f \in \mathcal{U}_1$:

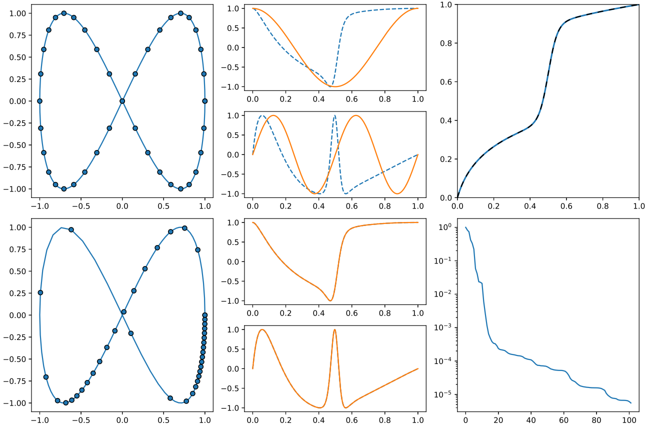
$$\kappa^{-1}(\mathcal{U}_1) \subseteq \mathrm{Diff}_\partial(\Omega).$$

From the lemma it follows that

$$\bigcup_{L\in\mathbb{N}} \underbrace{\kappa^{-1}(\mathcal{U}_1) \circ \cdots \circ \kappa^{-1}(\mathcal{U}_1)}_{L} = \mathrm{Diff}_{\partial,0}(\Omega)$$

where $\mathrm{Diff}_{\partial,0}(\Omega)$ is the connected component of the identity which is open in the whole group $\mathrm{Diff}_\partial(\Omega)$.

Recovering the (optimal) parametrization

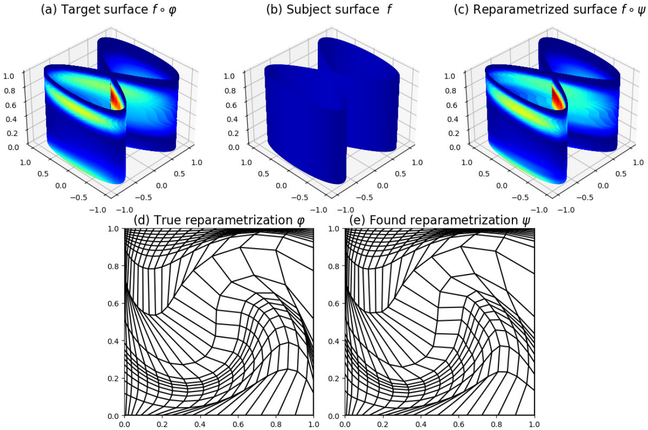


- (Left) The curve c to be reparametrized on top, and the target curve $c \circ \varphi$ below
- (Top right) The true reparametrization φ (dotted black) compared to the reparametrization ψ found by the algorithm.
- (Bottom right): The value of the loss function plotted against the iteration number of the weight updates, given relative to the initial error. Each iteration corresponds to one iteration of the BFGS algorithm including line-search.

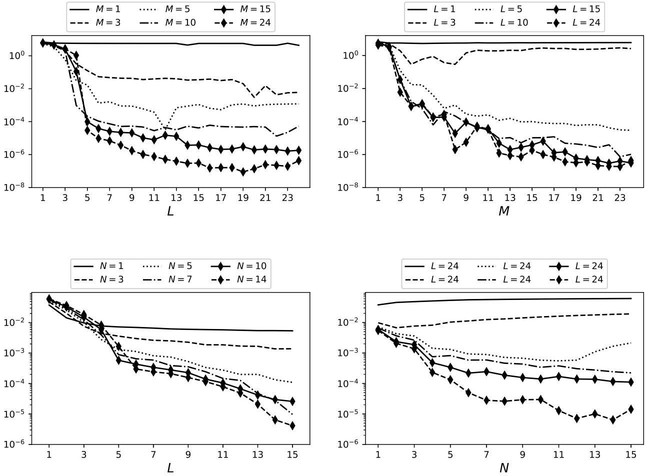
Example, cylinder

$$f(x,y) = [\sin(2\pi), \sin(4\pi x), y]^T, \quad f \circ \varphi$$

$$\varphi = \left[0.9x^2 + 0.1x,\ \frac{\log(20y+1)}{2\log(21)} + \frac{1+\tanh(29(y-0.5))}{4\tanh(10)}\right]$$

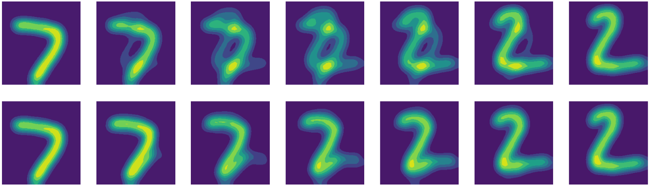


Convergence



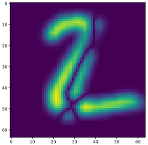
Data generation, optimal deformation

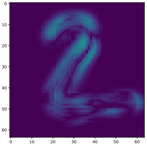
MNIST, matching of images, handwritten digits

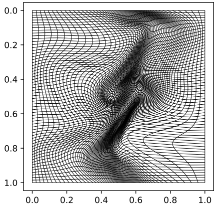


Top $\lambda(f_1, f_2, \tau) := \tau f_1 + (1-\tau) f_2$

Bottom $\gamma(f_1, f_2, \tau) := \tau f_1 + (1-\tau) f_2 \circ \varphi^*$, φ^* optimal reparametrization.







Learning ODEs from data

For learning the vector field of a Hamiltonian system $\dot{y} = X_H(y) = J\,\nabla H(y)$ we learn the Hamiltonian

$$H \approx H_\Theta$$

Assuming

$$\{\tilde{y}_i^0, \ldots, \tilde{y}_i^M\}_{i=1,\ldots,N}$$

are the N observed time trajectories of the flow of the vector field X_H that we want to learn.
Loss

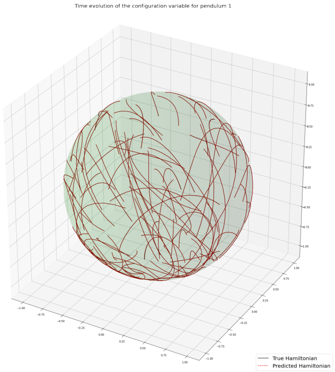
$$\mathcal{L}(\Theta) = \frac{1}{2n}\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\|y_i^j(\Theta) - \tilde{y}_i^j\|^2,$$

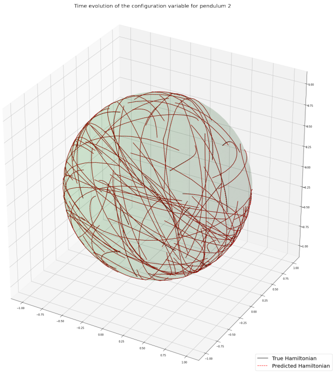
where

$$y_i^j(\Theta) = \Phi^h_{X_{H_\Theta}}(y_i^{j-1})$$

Architecture of the network: a recurrent neural network.

Mechanical systems with constraints

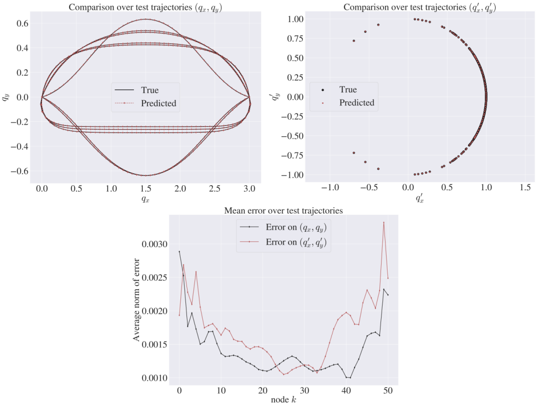




Comparison between 100 test trajectories obtained with the true Hamiltonian H and the predicted one H_Θ. To train H_Θ, a Lie group method is used. This gives $\mathcal{E}_1 = 2.65 \cdot 10^{-6}$ and a final training loss of $1.6 \cdot 10^{-9}$.

- EC, Leone, Murari and Owren, *Learning Hamiltonians of constrained mechanical systems*, J CAM, 2022

Efficient prediction of beam deformation aided by neural networks

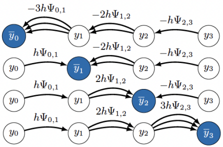


Data set splitting Training - test	Training accuracy	Test accuracy
10% - 10%	$2.383 \cdot 10^{-4}$	$7.784 \cdot 10^{-4}$
20% - 10%	$5.612 \cdot 10^{-5}$	$7.285 \cdot 10^{-5}$
40% - 10%	$7.104 \cdot 10^{-6}$	$9.275 \cdot 10^{-6}$
90% - 10%	$1.869 \cdot 10^{-6}$	$4.810 \cdot 10^{-6}$

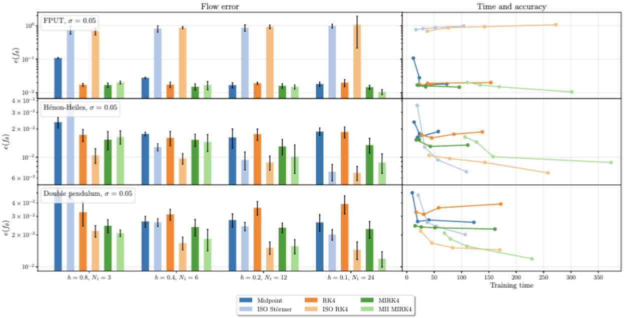
Table 8 Behaviour of the continuous network q_ρ^c tested on the *both-ends* data set with fewer training data points. The size of the training set varies, while that of the test set is fixed. The last row corresponds to the results in Figure 3.

- EC, Cokaj, Leone, Leyendecker, Murari, Owren, Sato, Stavole *Neural networks for the approximation of Euler's elastica*, 2023 arXiv

Learning Hamiltonians from noisy data: mean inverse integrator



- MII uses the group property of the (numerical) flow to produce and average over different approximations of the same value $y(t_n)$, reducing noise.
- MII is best combined with MIRK (inverse explicit).



- Noren, Eidnes, EC, *Learning Dynamical Systems from Noisy Data with Inverse-Explicit Integrators*, 2023.

References

- EC, Murari, Owren, Schönlieb and Sherry, *Dynamical systems based neural networks*, 2023, SISC
- EC, Leone, Murari and Owren, *Learning Hamiltonians of constrained mechanical systems*, J CAM, 2022
- EC, Cokaj, Leone, Leyendecker, Murari, Owren, Sato, Stavole *Neural networks for the approximation of Euler's elastica*, 2023 arXiv
- Noren, Eidnes, EC, *Learning Dynamical Systems from Noisy Data with Inverse-Explicit Integrators*, 2023.
- Sherry, EC, Ehrhardt, Murari, Owren and Schönlieb, *Designing Stable Neural Networks using Convex Analysis and ODEs*, 2024, Physica D.
- Arnold, EC, Cokaj, Owren, Tumiotto, *Contractivity of numerical integrators on Riemannian manifolds*, 2024, JCD.
- E. Celledoni, H. Glöckner, J. Riseth and A. Schmeding, *Deep learning of diffeomorphisms for optimal reparametrizations of shapes*, BIT, 2023.
- E. Celledoni, M. Eslitzbichler and A. Schmeding, *Shape analysis on Lie groups with applications in computer animation*, J Geometric Mechanics, 2016.
- E. Celledoni, S. Eidnes and A. Schmeding, *Shapes on homogeneous manifolds*, The Abel Symposium, 187-220, 2018.
- M. Eslitzbichler, *Modelling character motions on infinite-dimensional manifolds*, The Visual Computer, 2014.
- P. E. Lystad, *Signatures in Shape Analysis*, Master Thesis, NTNU, 2019.
- J. Riseth, *Gradient-based algorithms in shape analysis for reparametrisation of parametric curves and surfaces*, Master Thesis, NTNU, 2021.

Thank you for listening.