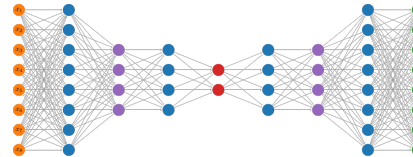
MAX-PLANCK-INSTITUT
FÜR PLASMAPHYSIK

Structure-Preservation and Manifold Optimization in Deep Learning for Scientific Computing



Benedikt Brantner[1,2]

Geometric Viewpoints on Learning, Data, and Neural Networks

LMU Munich, Thursday, September 24, 2026

The figures in this deck were "GIF-ed up" with `claude`.

[1] Max–Planck–Institut für Plasmaphysik, *Geometric Numerical Integration and Reduced Complexity Modelling* — ipp.mpg.de/4105128/gspm
[2] Technische Universität München, Zentrum Mathematik

The Traffic Runs Both Ways

1 · networks that solve differential equations
and are made to respect their structure *(Parts I and II)*

Machine learning



neural networks, attention,
gradient-based training

Scientific computing



differential equations,
geometric integrators, FEEC

2 · numerics that improves the learning machinery
optimisation on manifolds — a vision transformer on MNIST *(Part III)*

The Mathematics It Takes

Differential geometry *Part III*

- manifolds, tangent spaces, retractions
- Lie groups $SO(N)$, Lie algebras $\mathfrak{g}$
- homogeneous spaces $St(n, N)$, $Gr(n, N)$
- $\mathfrak{g} = \mathfrak{g}^{\text{ver}} \oplus \mathfrak{g}^{\text{hor}}$, global sections $\lambda(Y)$
- geodesics, parallel transport

Numerics *Parts I & III*

- function bases: splines per patch, degree p
- FEEC: a *sub*complex, commuting π_h
- exp: Taylor and Padé kernels
- scaling & squaring, Cayley, QR completion
- backward error analysis

`GeometricOptimizers.jl`

Geometric machine learning *Part II*

- SympNets
- symplectic autoencoders

Structure preservation *Parts I & II*

a symplectic form

- ω, $\dot{z} = \mathbb{J}\nabla_z H$, symplectic flows
- Poincaré invariants, non-squeezing
- symplectic integrators, modified H_h

a de Rham complex

- $d \circ d = 0$, so $\operatorname{im} d \subseteq \ker d$
- cohomology $\dim \mathcal{H}^k = b_k(\Omega)$
- harmonic fields $\ker \operatorname{curl}_h / \operatorname{im} \operatorname{grad}_h$

a symmetry

- Noether, discrete momentum maps

- what is preserved, the numerics that preserves it, and the geometry this leads to.

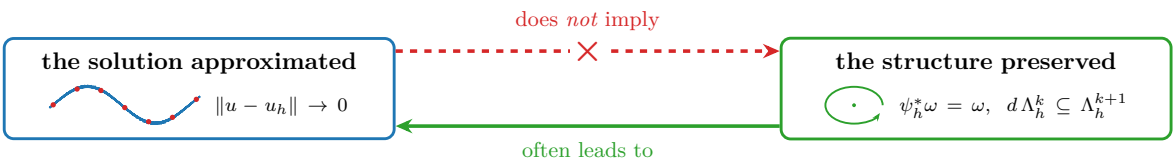
Outline

I. Why Structure Preservation? Lessons from Scientific Computing

II. Structure-Preserving Neural Network Architectures

III. Training: Optimization on Manifolds

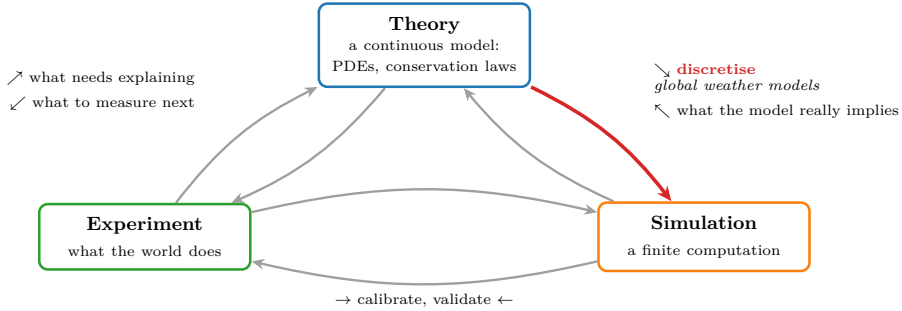
IV. Open Questions



Approximating the *solution* well does not imply structural properties, but structural properties often lead to good solutions.

Why Structure Preservation? Lessons from Scientific Computing

Numerics



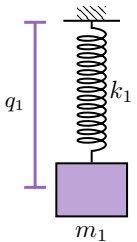
- *"Mathematics is a part of physics. Physics is an experimental science, a part of natural science. Mathematics is the part of physics where experiments are cheap."* (V. I. Arnold, *On Teaching Mathematics*, 1997); and simulation is an indispensable tool for mathematics,
- **numerical analysis** is the study of the red arrow: replace a continuous model by a finite computation, and account for what the replacement costs,
- the **potency** of a numerical method can be quantified through the error $\|u - u_h\| \to 0$ — or through its structural properties. **This talk is about the second.**

A Deliberately Trivial Problem

- the harmonic oscillator, $k = \frac{1}{2}$, $m = 1$, $q(0) = 1$, $\dot{q}(0) = 0$:

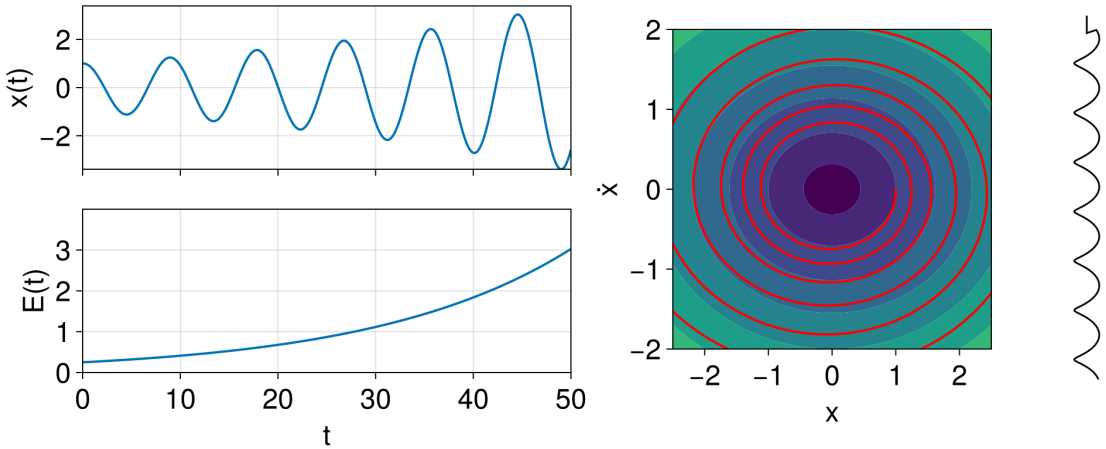
$$m\ddot{q}(t) = -kq(t), \qquad H(q,p) = \frac{p^2}{2m} + \frac{k}{2}q^2.$$

- two methods, both first order, both explicit, both costing one force evaluation per step:
  1. explicit Euler,
  2. symplectic Euler.
- over one step you cannot tell them apart.



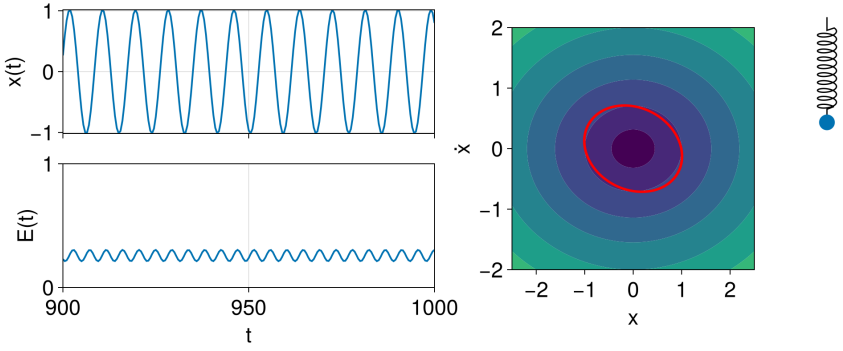
Structure becomes crucial on long time scales.

Explicit Euler $(h = 0.1)$



- the amplitude grows, the energy grows: the numerical trajectory spirals outwards forever.

Symplectic Euler $(h = 0.1)$



- the energy oscillates in a bounded band; explicit Euler had already left the plot at $t = 50$,
- same order, same cost, same one force evaluation per step. The difference is **geometry**.
- a symplectic method is — up to exponentially small terms — the exact flow of a modified Hamiltonian $H_h = H + h^p H_{p+1} + \cdots$, which it conserves exactly, so $|H(z_n) - H(z_0)| = \mathcal{O}(h^p)$ for $t \lesssim e^{c/h}$ (*backward error analysis*, Hairer et al., 2006; appendix).

The Same Story for PDEs: the Problem

- **the object is a complex, not an equation:** a sequence of spaces joined by differential operators, with $d \circ d = 0$,
- **the domain:** the pretzel of Psydac (Güçlü et al., 2022) — 11 spline patches glued along 13 interfaces (an interface = one shared edge), one piece, three holes,
- **the answer is fixed before a mesh exists:** $\dim \mathcal{H}^k = b_k(\Omega)$. Here $b_1 = 3$,



three loops, none of them contractible

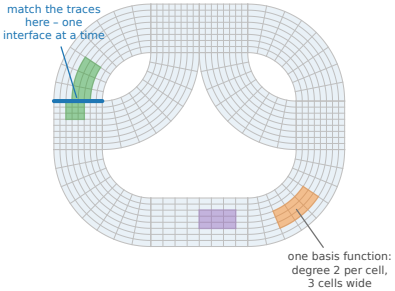
- **traditionally spaces are chosen for each patch:** degree p per cell, continuity where the weak form asks, vector unknowns componentwise; nothing in it mentions the sequence, $d \circ d$, or the topology of Ω[1],
- **solution:** choose the spaces *together* — **FEEC**, at the end of Part I.

What is computed: the Hodge Laplacian on 1-forms, $\Delta_h = \mathrm{curl}^* \mathrm{curl} + \mathrm{grad}\,\mathrm{grad}^*$, as $\Delta_h u = \lambda M u$ (M the mass matrix); its kernel is the harmonic fields $\mathcal{H}^1_h = \ker \mathrm{curl}_h / \mathrm{im}\,\mathrm{grad}_h$. **Expected: exactly $b_1 = 3$, for any mesh.**

[1] Boffi, 2010; Bossavit, 1990.

Why the Usual Recipe Cannot See a Hole

how the spaces are usually chosen



what the answer depends on

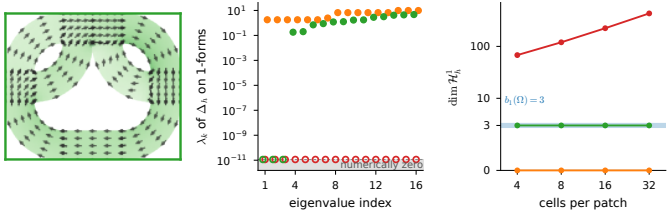


$$\dim \mathcal{H}^1 = b_1(\Omega) = 3$$

- **left:** every condition is local — one cell, one interface,
- **right:** b_1 is a property of the whole domain,
- **refining** shrinks the blobs on the left and changes nothing on the right — so the spaces have to be chosen together.

The Same Story for PDEs: the Evidence

- the same complex, built three times, differing only in which rungs are tied across the 13 interfaces (*tied* = shared coefficients there; nothing tied = a separate spline per patch, not even continuous),
- the middle one *is* the standard construction: glue Λ_h^0, take Λ_h^1 componentwise,
- measured is the cohomology: against $b_0 = 1$, $b_1 = 3$.



Λ_h^0 *and* Λ_h^1 tied Λ_h^0 only nothing tied degree 2, 4 to 32 cells per patch. *left* a harmonic field · *middle* the 16 smallest λ_k · *right* how many are zero: **3** v. **432** and rising v. **0** — too many, too few: *the two failures point opposite ways*.

Why good splines are not enough. approximation power is a property of *one* space and is local; cohomology is a property of the *sequence* and is global. Build Λ_h^1 componentwise and $\operatorname{grad} \Lambda_h^0 \not\subset \Lambda_h^1$.

What the Good Method Preserved: Hamiltonian Systems

- **Symplectic ODEs**:

$$\dot{z} = f(z) = \mathbb{J}\nabla_z H \text{ where } \mathbb{J} = \begin{pmatrix} \mathbb{O} & \mathbb{I} \\ -\mathbb{I} & \mathbb{O} \end{pmatrix},$$

  for a *Hamiltonian* $H : \mathrm{Phase\,Space} \to \mathbb{R}$.

- **Symplectic Flow** φ:

$$(\nabla_z \varphi)^T \mathbb{J} \nabla_z \varphi = \mathbb{J}.$$

- a *symplectic integrator* is a map ψ_h with the same property — it is a legitimate *Hamiltonian* evolution, just not of H.

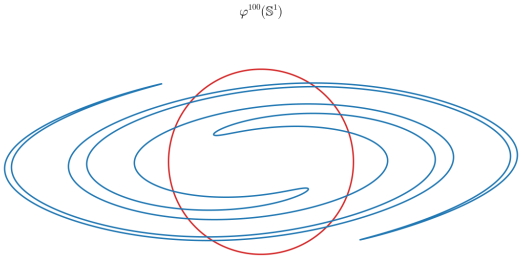


Figure 1: Example of a symplectic map φ in two dimensions.

Symplecticity is Stronger than Energy Conservation

- energy preservation confines the evolution to a $(2n-1)$-dimensional submanifold — but gives no other guarantee,
- preservation of symplecticity $\Longrightarrow$
  - conservation of all Poincaré invariants,
  - Gromov's non-squeezing theorem.

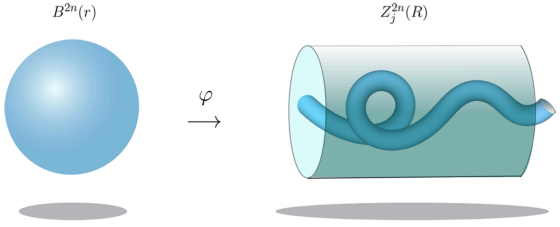
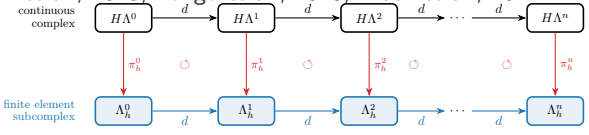


Figure 2: Visualization of Gromov's non-squeezing theorem. Picture is taken from Qin, 2024.

Null Spaces of Differential Operators $\Longrightarrow$ FEEC

- the sequence the pretzel was discretized on:
$$\Lambda_h^0 \xrightarrow{\text{grad}} \Lambda_h^1 \xrightarrow{\text{curl}} \Lambda_h^2, \qquad d \circ d = 0,$$
- what failed on it was a null space: $\operatorname{im} d \subseteq \ker d$, and the cohomology $\ker d / \operatorname{im} d$ is an invariant of the *topology* of the domain: $\dim \mathcal{H}^0 = b_0 = 1$, $\dim \mathcal{H}^1 = b_1 = 3$,
- **FEEC** (Arnold et al., 2006): make the discrete spaces a subcomplex, joined to the continuous one by projections that commute with d — then the discrete cohomology *is* the continuous one,
- the failure mode is not drift but *instability*: spurious modes, *more* of them the finer the grid. Networks are already built on this: Kinch et al., 2025; Longo et al., 2023; Trask et al., 2022.



$\pi_h^{k+1} d = d \pi_h^k \quad \Longrightarrow \quad$ the discrete complex inherits the cohomology

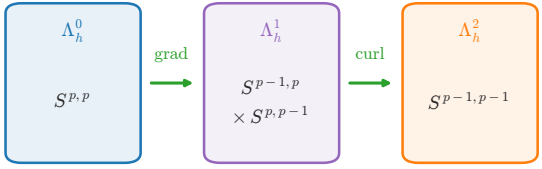
$H\Lambda^k$ is the *domain of* d.

FEEC ensures that $d\,\Lambda_h^k \subseteq \Lambda_h^{k+1}$

every rung conforming, and tied *compatibly* across every interface

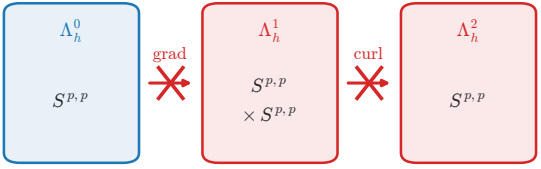
FEEC Ensures $d\,\Lambda_h^k \subseteq \Lambda_h^{k+1}$

the ladder FEEC picks



differentiating drops the degree in the direction differentiated, so the image is exactly the next space

the componentwise choice



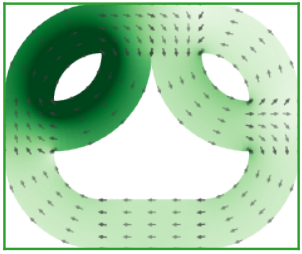
every space is at least as accurate, and $\operatorname{grad} S^{p,p} \not\subseteq S^{p,p} \times S^{p,p}$

left — the spaces picked *together*; right — one space per unknown, every space at least as accurate.

Bookkeeping, not approximation power — and settled before a single matrix is assembled.

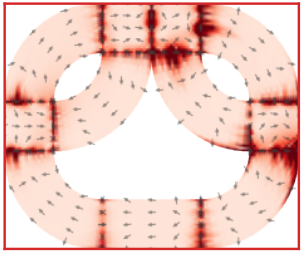
What a Broken Complex Actually Produces

compatible: one of the three



spread over the domain, circulating round a hole; conforming to 10^{-16}

Λ_h^0 tied only: one of the 224



living on the interfaces: the gradient of a potential that jumps across them

- same eigensolver, same mesh, same degree; both panels are an eigenfunction of Δ_h with eigenvalue 0 — a harmonic field, an element of $\ker \Delta_h$,
- left — the compatible complex: a global object, circulating round a hole, conforming to 10^{-16}. There are exactly three,
- right — Λ_h^0 tied, Λ_h^1 broken: the field lives on the thirteen interfaces, a gradient inside each patch of a potential that jumps across them. One of 224 here, 432 at $n = 32$ — *there is no mesh at which this panel improves*.

The Principle

for ODEs

*"[Geometric numerical integration] deals with numerical integrators that **preserve geometric properties of the flow** of a differential equation, and it explains how **structure preservation leads to improved long-time behaviour**."*

(Hairer, Lubich & Wanner, *Acta Numerica* **12**, 2003, abstract)

*"With the interest in computations over long time intervals one discovered that certain methods **reproduce the qualitative behavior of the exact flow** much better than others."*

(Hairer, *Challenges in Geometric Numerical Integration*, 2014, §1)

for PDEs

*"An important but insufficiently appreciated point is that approximability and consistency together with well-posedness of the continuous problem **do not imply stability**."*

*"[...] stability [...] can be obtained by designing methods which are **compatible** with these structures in the sense that they **reproduce or mimic them (not just approximate them)** on the discrete level."*

(Arnold, Falk & Winther, *Acta Numerica* **15**, 2006, §1)

Reproduce the structure. Do not merely approximate it!

(Arnold et al., 2006; Hairer, 2014; Hairer et al., 2003)

The Principle, in Three Rows

continuous structure	**discrete counterpart**	**what it buys**
symplectic form ω	symplectic integrator	bounded energy error, exponentially long times
de Rham complex	commuting cochain projection	stability, no spurious modes
symmetry (Noether)[2]	discrete symmetry of L_d	exactly conserved discrete momentum

And when the discrete map is a *neural network*?

[2]Discrete Noether: Marsden and West, 2001; Hairer et al., 2006, Ch. VI. Learned, symmetry and all: Dierkes et al., 2023; Lishkova et al., 2023; Offen and Ober-Blöbaum, 2022. In ML the same idea is equivariance: Bronstein et al., 2021; Cohen and Welling, 2016.

Structure-Preserving Neural Network Architectures

Neural Networks ...

- ...offer a way of approximating functions (like polynomials, Fourier series, B-splines, ...),
- ...consist in their simplest form of an affine transformation and a nonlinearity:
  1. affine transformation: $x \mapsto Ax + b$,
  2. nonlinearity σ: tanh, sin, cos, $x \mapsto \frac{1}{1+e^{-x}}$, $x \mapsto \max(0, x)$, ...
- neural networks are composed of many of these *layers*: $\psi^{\mathcal{NN}} = \psi_n \circ \cdots \circ \psi_1$ where $\psi_i(x) = \sigma(Ax + b)$.

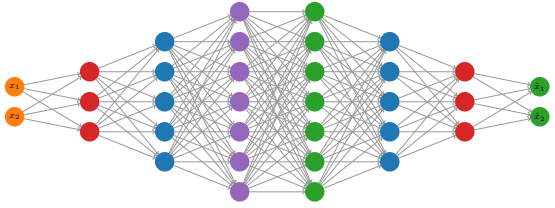


Figure 3: A neural network with input x and output $\hat{x}$.

- universal approximation theorem[3].

[3] Hornik et al., 1989.

The Reframing

- a network trained to advance a state by h *is* a one-step method,
- so every argument from Part I applies to it verbatim: approximate the analytic solution of $\dot{z} = f(z) = \mathbb{J}\nabla_z H$ ...
- ...and ask which *structure* the approximation preserves.

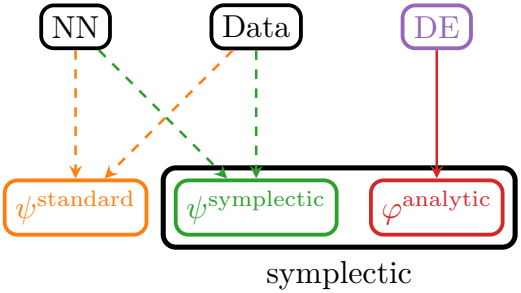


Figure 4: Structure-preserving networks in one picture.

Hamiltonian Neural Networks

- *discovery* of a Hamiltonian from data[4]
- if vector-field data are available:

$$\mathcal{L}_{\text{HNN}} = \sum_t \left|\left| \frac{\partial H_\theta}{\partial p}(q^{(t)}, p^{(t)}) - \dot{q}^{(t)} \right|\right|_2 + \left|\left| \frac{\partial H_\theta}{\partial q}(q^{(t)}, p^{(t)}) + \dot{p}^{(t)} \right|\right|_2$$

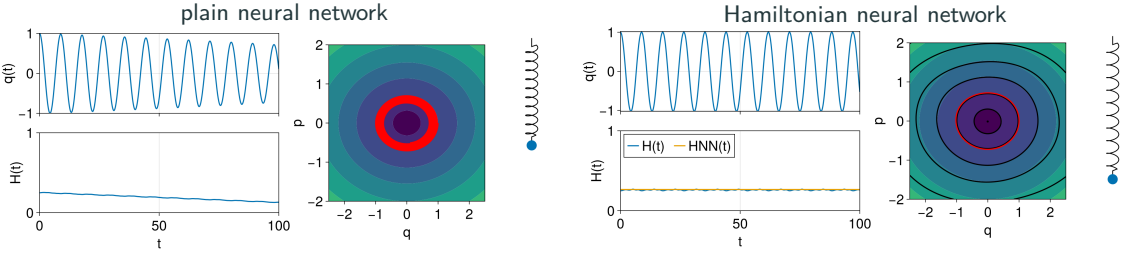
- if phase space data are available:

$$\mathcal{L}_{\text{HNN}} = \sum_t \left|\left| \frac{\partial H_\theta}{\partial p}\left(q^{(t)}, p^{(t+1)}\right) - \frac{q^{(t+1)} - q^{(t)}}{\Delta t} \right|\right|_2 + \sum_t \left|\left| \frac{\partial H_\theta}{\partial q}\left(q^{(t)}, p^{(t+1)}\right) + \frac{p^{(t+1)} - p^{(t)}}{\Delta t} \right|\right|_2$$

- the structure is in what is learned: a scalar H_θ, integrated afterwards with a method of our choosing.

[4] Greydanus, S., Dzamba, M., & Yosinski, J. (2019). **Hamiltonian neural networks.** *Advances in neural information processing systems, 32.*

The Harmonic Oscillator, Learned $(h = 0.1)$

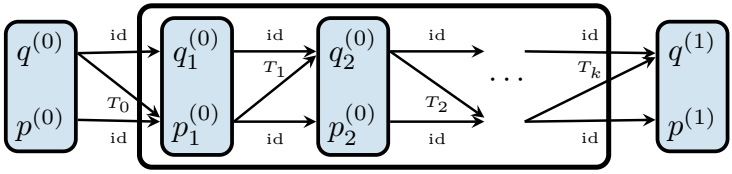


- same data, same parameter budget, same training. The difference is the inductive bias, not the fit.

	derived (Part I)	**learned** (Part II)	
no structure	explicit Euler	plain neural network	the energy drifts
structure	symplectic Euler	HNN / SympNet	the energy stays bounded

SympNets: Symplectic *by Construction*

- SympNets can approximate arbitrary symplectic maps[5] and ...
- ...therefore also the solution of Hamiltonian systems (for a fixed time step).



- structurally, a SympNet *is* a splitting method: symplectic Euler applied to $H(q,p) = T(p) + U(q) = a_1^T \Sigma(K_1 x + b_1) + a_2^T \Sigma(K_2 x + b_2)$ — with learned T and U.

[5] Jin, P., Zhang, Z., Zhu, A., Tang, Y., & Karniadakis, G. E. (2020). **Sympnets: Intrinsic structure-preserving symplectic networks for identifying hamiltonian systems.** *Neural Networks*, *132*, 166–179.

Generalized Hamiltonian Neural Networks

Since a SympNet *is* a splitting method, we can *generalize* it: more expressive networks for T and U, and any integrator we like.

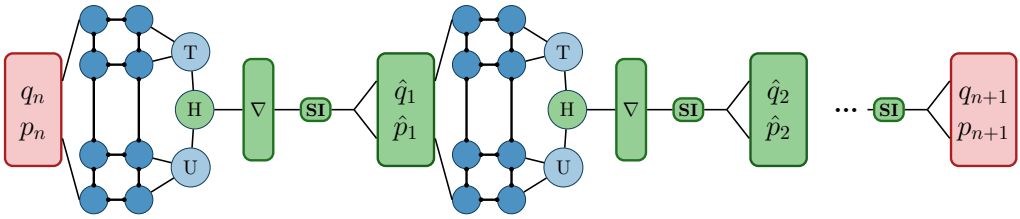


Figure 5: GHNNs combine the HNN and SympNet approach. Figure adjusted from Horn et al., 2025.

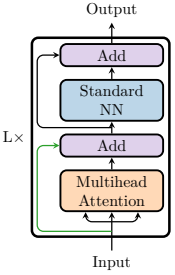
- the architecture and the integrator become separate design choices — which is what lets us reach settings SympNets cannot (see below).

Beyond One Step: Transformers

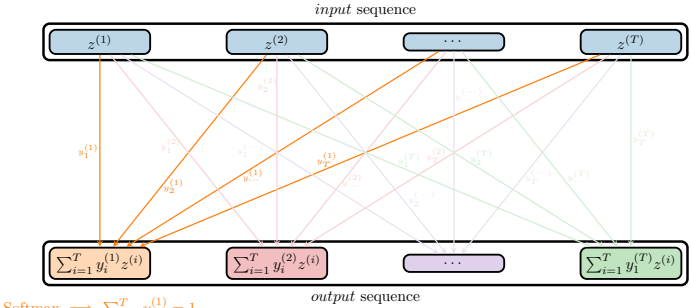
- can process time series,
- attention layer[6]:

$$Z \mapsto Z\mathrm{softmax}(Z^T AZ)$$

where $[\mathrm{softmax}(C)]_{ij} = \dfrac{e^{c_{ij}}}{\sum_i c_{ij}}$.



- ChatGPT,
- learned convex reweighting of input vectors:



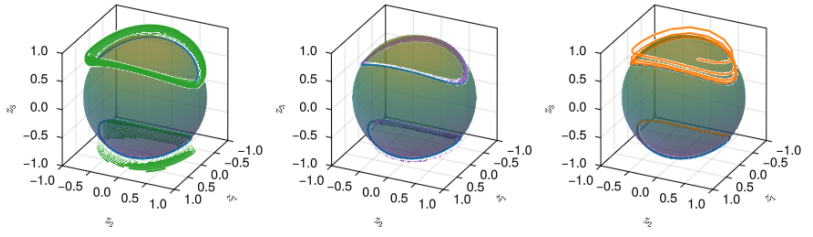
Softmax $\implies \sum_{i=1}^T y_i^{(1)} = 1$

- a multistep method with a *learned*, state-dependent set of coefficients.

[6] Vaswani et al., 2017.

Making Attention Structure-Preserving

- the softmax acts **column-wise** $\implies$ it cannot preserve a symplectic form on $\mathbb{R}^{2n} \times \cdots \times \mathbb{R}^{2n}$,
- replace it: with $\sigma(C) = \mathrm{Cayley}(\texttt{lower_triangular_asymmetrize}(C))$ the reweighting matrix $\Lambda(Z) = \sigma(Z^T A Z)$ is orthonormal $\implies$ *volume-preserving* attention[7],
- or build the attention itself as a multistep symplectic Euler step $\implies$ *symplectic* attention,
- and the feed-forward block: standard NN $\to$ volume-preserving NN[8] / SympNet.



Rigid body, two validation trajectories:

- implicit midpoint *(reference)*
- VP feedforward *(left)*
- VP transformer *(middle)*
- standard transformer *(right)*

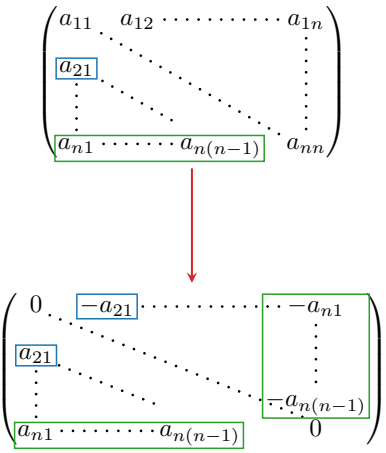
[7] B. et al., 2025.
[8] Bajārs, 2023.

Structure-Preserving Attention: the Construction

- we replace the softmax activation function:

$$\sigma(C) = \mathrm{Cayley}(\texttt{lower_triangular_asymmetrize}(C))$$

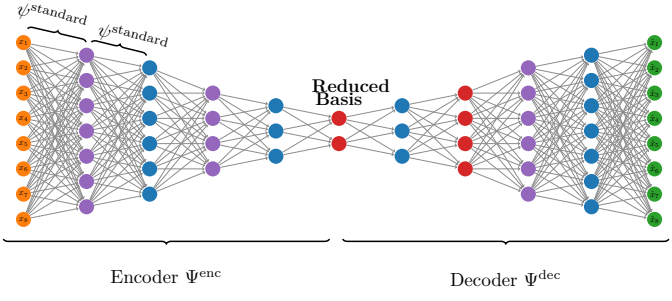
- $[\texttt{lower_triangular_asymmetrize}(C)]_{ij} = \begin{cases} c_{ij} & \text{if } i > j \\ -c_{ji} & \text{if } i < j \\ 0 & \text{else,} \end{cases}$
- $\mathrm{Cayley}(Y) = (\mathbb{I}_T - Y)(\mathbb{I}_T + Y)^{-1}$ maps skew-symmetric matrices to orthonormal ones,
- so $\Lambda(Z) = \sigma(Z^T A Z)$ is orthonormal.



Reduced Order Modeling: Standard Autoencoders

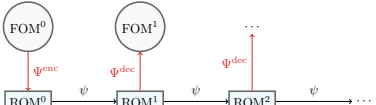
- autoencoders: $(\Psi^{\text{enc}}, \Psi^{\text{dec}})$,
- goal: learn low-dimensional representation of data,
- reconstruction loss:

$$Loss : z \mapsto ||z - \Psi^{\text{dec}} \circ \Psi^{\text{enc}}(z)||.$$



With **manifold Galerkin projection**[9]:

$$f^{\text{ROM}}(\tilde{z}) = (\nabla_{\tilde{z}} \Psi^{\text{dec}})^{+} f^{\text{FOM}}(\Psi^{\text{dec}}(\tilde{z})).$$

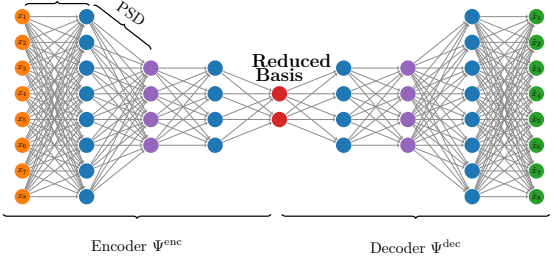


[9] Lee and Carlberg, 2020.

Symplectic Autoencoders

- reconstruction loss:

$$Loss : z \mapsto ||z - \Psi^{\text{dec}} \circ \Psi^{\text{enc}}(z)||.$$



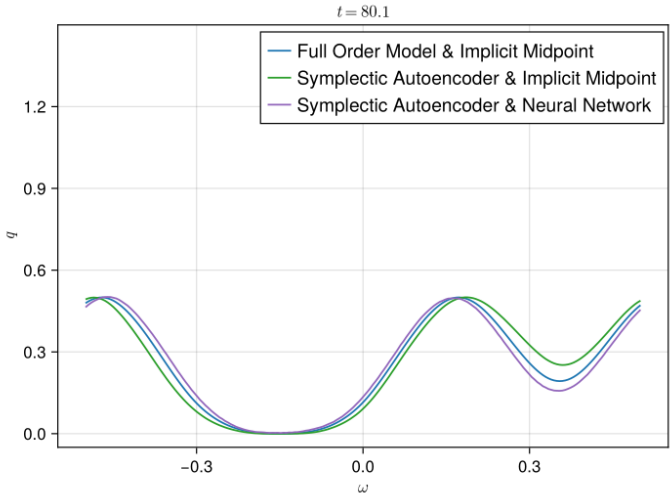
With **symplectic manifold Galerkin projection**[10]:

$$f^{\text{FOM}}(z) = \mathbb{J}_{2N} \nabla_z H$$
$$f^{\text{ROM}}(\tilde{z}) = \mathbb{J}_{2n} \nabla_{\tilde{z}} (H \circ \Psi^{\text{dec}}).$$

- SympNets[11] & Proper Symplectic Decomposition[12] (PSD) $\implies$ symplectic autoencoders[13],
- note: the encoder's weights now have to satisfy $A^T A = \mathbb{I}$ — remember this for Part III.

[10] Buchfink et al., 2023.
[11] Jin et al., 2020.
[12] Peng and Mohseni, 2016.
[13] B. and Kraus, 2023.

Applied to the Toda Lattice



Simulation times on CPU:

- Full Order Model & Implicit Midpoint: **114 s**,
- Symplectic Autoencoder & Implicit Midpoint: **241 s** ($2\times$ slowdown),
- Symplectic Autoencoder & Neural Network: **0.295 s** ($300\times$ speedup).
- a traditional integrator in the online phase jeopardizes the entire advantage gained in the offline phase — which is why the *propagator* has to be a network too, and therefore has to be structure-preserving.

Benefits, Limitations, and What to Do About Them

What We Gain, and What We Pay

What we gain

- bounded error over long horizons instead of drift,
- output that is physically admissible by construction, not merely close,
- far less data: the structure does work that data would otherwise have to do,
- useful extrapolation beyond the training window,
- a ROM whose online phase is actually fast.

What we pay

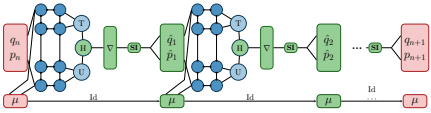
- a constrained hypothesis class: a SympNet is a splitting method,
- implicit layers and inverse/exponential maps can be expensive,
- we have a structure theory but no matching approximation theory: we know the map is symplectic, not how well it approximates,
- constrained weights make training harder (Part III).

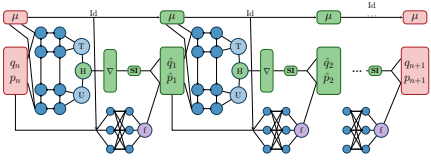
More General Systems

1. **Generalize the architecture, not the constraint.** A GHNN keeps exact symplecticity but lets the integrator and the networks for T, U be chosen freely.

2. **Widen the setting.** Parametrised and forced GHNNs extend the construction to non-autonomous and dissipative problems (B. & Tyranowski, 2026).

3. **Keep the structure in the architecture, not in the loss.** A PINN-style penalty is soft. It is a preference, not a constraint.





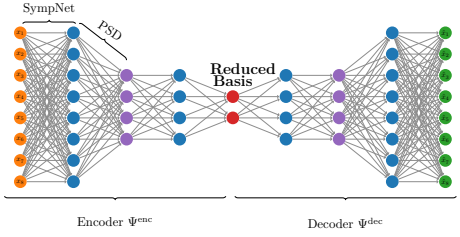
parametrised and forced GHNNs

Exact structure sometimes puts the weights on a **manifold** ($A^T A = \mathbb{I}$, ...) — and moves the difficulty into the **optimizer**. That is Part III.

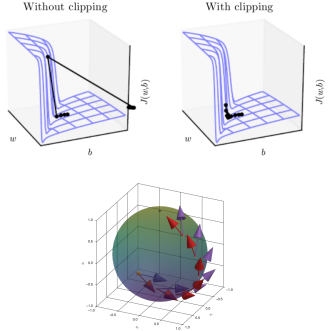
Training: Optimization on Manifolds

Where the Manifold Comes From

1. to make training of some networks possible at all: the symplectic autoencoder, whose encoder weights must satisfy $A^T A = \mathbb{I}_n$,
2. to mitigate the vanishing/exploding gradient problem in CNNs, ResNets and transformers.



1. PSD layers: $St(n, N)$ by construction.



2. clipping (Goodfellow et al., 2016) vs. optimization on a compact space.

these are two independent reasons: one from structure preservation, one from plain deep learning.

Typical Neural Network Optimizers

Algorithm 1 High-level optimization algorithm for neural networks

Require: time t, weights $Y^{(t)}$, `state`, $\mathcal{O}$-parameters Ξ;

$\Delta^{(t)} \leftarrow \nabla_{Y^{(t)}} L$ ▷ compute the Euclidean gradient.

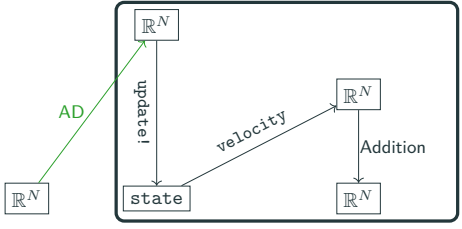
$\texttt{state} \leftarrow \texttt{update}(\texttt{state}, \Delta^{(t)}, t, \Xi)$ ▷ update state based on $\Delta^{(t)}$.

$W^{(t)} \leftarrow \texttt{velocity}(\texttt{state}, \Xi)$ ▷ compute *final velocity*.

$Y^{(t+1)} \leftarrow Y^{(t)} + W^{(t)}$ ▷ add *final velocity* to $Y^{(t)}$.

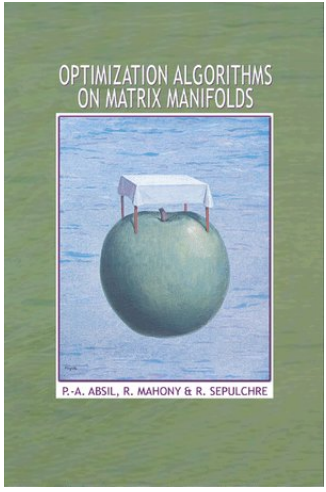
Replace:

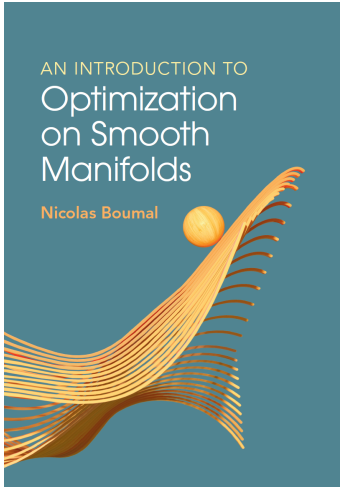
- $Y \in \mathcal{M}$,
- Euclidean Gradient $\rightarrow$ Riemannian Gradient $\implies$ $\operatorname{grad}_{Y^{(t)}} L \in T_{Y^{(t)}}\mathcal{M}$,
- Addition $\rightarrow$ Geodesic/Retraction.



`optimization_step`$(\mathsf{o}, ::\mathbb{R}^N, ::\mathbb{R}^N)$

So Why Not Use an Existing Manifold Optimizer?





- a wide array of manifold optimizers exists[14], but ...
- ...they usually do not encompass the idiosyncratic optimizers that deep learning actually uses — Adam above all[15].

[14] Absil et al., 2008; Boumal, 2023.
[15] Kingma and Ba, 2014.

The Actual Obstruction

- updating the state:

$$\mathcal{B}_1^{\texttt{state}} \leftarrow \frac{\beta_1 - \beta_1^t}{1-\beta_1^t}\mathcal{B}_1^{\texttt{state}} + \frac{1-\beta_1}{1-\beta_1^t}\mathcal{B}_t,$$

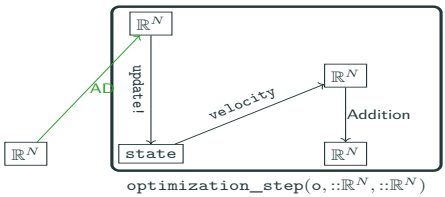
$$\mathcal{B}_2^{\texttt{state}} \leftarrow \frac{\beta_2 - \beta_2^t}{1-\beta_2^t}\mathcal{B}_2^{\texttt{state}} + \frac{1-\beta_2}{1-\beta_2^t}\mathcal{B}_t \odot \mathcal{B}_t,$$

$\implies$ involves operations ($\odot$, elementwise $\sqrt{\cdot}$) that lack a clear geometric interpretation!

- computing the velocity:

$$W^{(t)} \leftarrow \texttt{velocity}(\texttt{state}, \Xi) = -\eta \mathcal{B}_1^{\texttt{state}} / \sqrt{\mathcal{B}_2^{\texttt{state}} + \delta},$$

- and the state is carried *across* iterations — but $T_Y\mathcal{M}$ moves with Y.



The Actual Obstruction

- updating the state:

$$\mathcal{B}_1^{\texttt{state}} \leftarrow \frac{\beta_1 - \beta_1^t}{1-\beta_1^t}\mathcal{B}_1^{\texttt{state}} + \frac{1-\beta_1}{1-\beta_1^t}\mathcal{B}_t,$$

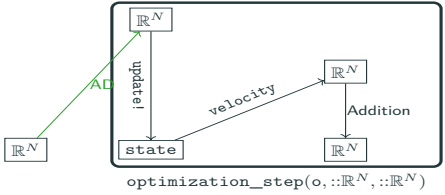
$$\mathcal{B}_2^{\texttt{state}} \leftarrow \frac{\beta_2 - \beta_2^t}{1-\beta_2^t}\mathcal{B}_2^{\texttt{state}} + \frac{1-\beta_2}{1-\beta_2^t}\mathcal{B}_t \odot \mathcal{B}_t,$$

$\implies$ involves operations ($\odot$, elementwise $\sqrt{\cdot}$) that lack a clear geometric interpretation!

- computing the velocity:

$$W^{(t)} \leftarrow \texttt{velocity}(\texttt{state}, \Xi) = -\eta \mathcal{B}_1^{\texttt{state}} / \sqrt{\mathcal{B}_2^{\texttt{state}} + \delta},$$

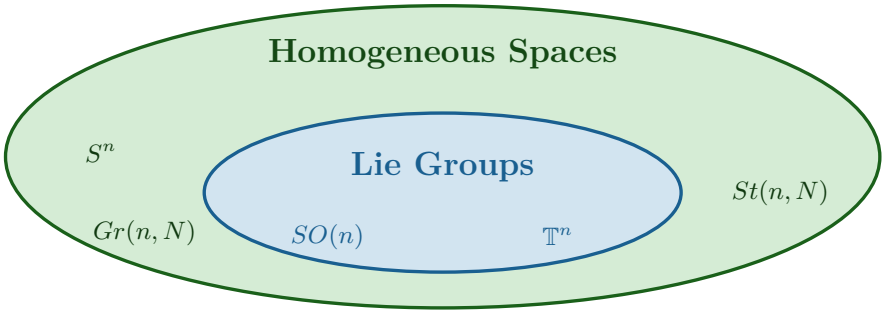
- and the state is carried *across* iterations — but $T_Y\mathcal{M}$ moves with Y.



The state has to be defined globally — in one fixed vector space.

Lie Groups & Homogeneous Spaces

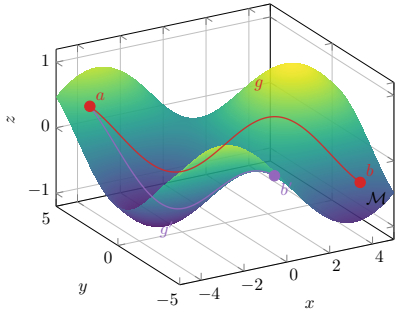
	Lie Group	**Homogeneous Space**
Definition	Manifold with group structure.	Manifold with transitive group action.
Identity	Distinguished identity element $\mathbb{I}$.	A *hand-picked* E.
Relationship	Trivially a homogeneous space.	Not necessarily a Lie group.



Homogeneous Spaces

"Es ist eine Mannigfaltigkeit und in derselben eine Transformationsgruppe gegeben; ***man soll die der Mannigfaltigkeit angehörigen Gebilde hinsichtlich solcher Eigenschaften untersuchen, die durch die Transformationen der Gruppe nicht geändert werden****."*
(Klein, *Erlanger Programm*, 1872, §I)[16]

- a manifold $\mathcal{M}$ on which a Lie group G acts **transitively** $\implies$ for any $a, b \in \mathcal{M}$ there is a $g \in G$ with $g \cdot a \equiv ga = b$,
- Stiefel manifold: $\{A \in \mathbb{R}^{N\times n} : A^T A = \mathbb{I}_n\}$, with $G = O(N)$ — exactly the constraint that showed up twice: (i) vanishing gradients in deep transformers, (ii) the symplectic autoencoder.



The same move as in Part I: name the group, keep what it leaves invariant.

[16] Cartan, 1930; Klein, 1872.

A Concrete Example: $SO(N)$ vs. $St(n, N)$

Special Orthogonal Group $SO(N)$

$$A \in \mathbb{R}^{N\times N} \quad \text{s.t.} \quad A^T A = \mathbb{I}_N$$

Identity Element:

$$\mathbb{I}_N$$

Lie Algebra ($\mathfrak{g}$):

$$\mathfrak{g} = \{B \in \mathbb{R}^{N\times N} \text{ s.t. } B^T = -B\}$$

Stiefel Manifold $St(n, N)$

$$A \in \mathbb{R}^{N\times n} \quad \text{s.t.} \quad A^T A = \mathbb{I}_n$$

Distinguished Element:

$$E = \begin{bmatrix} \mathbb{I}_n & \mathbb{O}^T_{(N-n)\times n} \end{bmatrix}^T$$

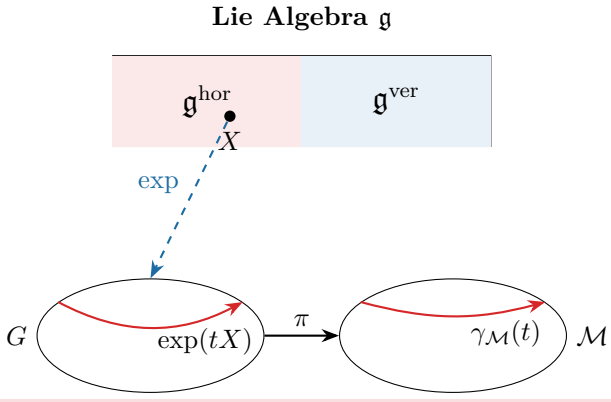
Horizontal Subspace ($\mathfrak{g}^{\text{hor}}$):

$$\mathfrak{g}^{\text{hor}} = \left\{ \begin{pmatrix} A \underbrace{\begin{bmatrix} B \\ \end{bmatrix}}_{n} & \underbrace{\begin{bmatrix} -B^T \\ 0 \end{bmatrix}}_{N-n} \end{pmatrix} \right\}.$$

Geodesics on Homogeneous Spaces

Algebraic Structure

	$\mathfrak{g}$	$\mathfrak{g}^{\text{hor}}$
Closure [,]	✓	✗
Represents	$T_{\mathbb{I}}G$	$T_E\mathcal{M}$

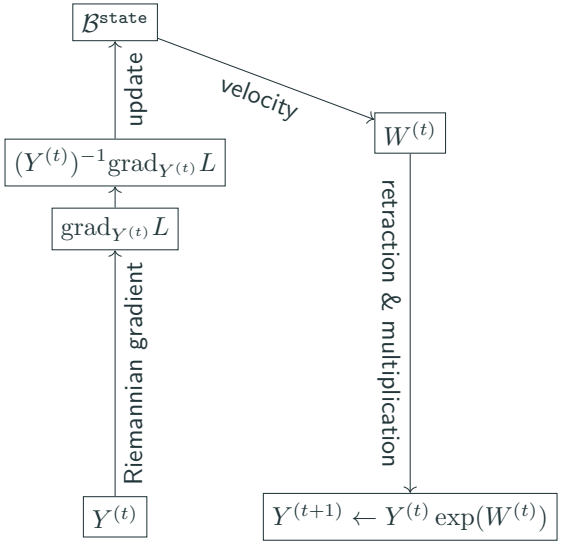


Geodesics on $\mathcal{M}$ are projections of exponentials from $\mathfrak{g}^{\text{hor}}$: $\gamma_{\mathcal{M}}(t) = \exp(tX) \cdot E$[17]

[17] Nomizu, 1954; O'neill, 1983.

Step 1 — Lie Groups

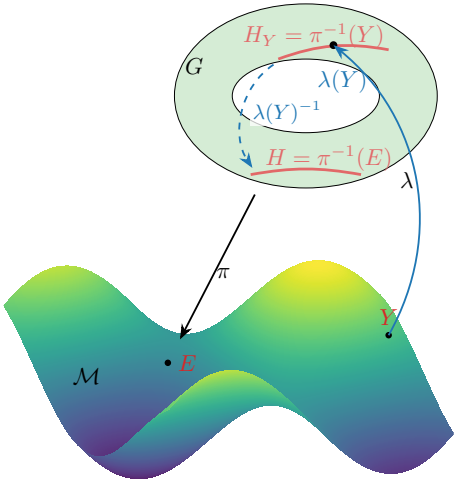
- Lie algebra $\mathfrak{g}$ as a *global tangent space*[18] $\implies$
  1. map $T_{Y^{(t)}}G$ to $\mathfrak{g}$,
  2. update the state $\mathcal{B}^{\texttt{state}}$ in $\mathfrak{g}$,
  3. compute *final velocity* $W^{(t)}$ in $\mathfrak{g}$,
  4. update weights with a *retraction* applied to $W^{(t)}$.
- $(Y^{(t)})^{-1}\mathrm{grad}_{Y^{(t)}}L$, $\mathcal{B}^{\texttt{state}}$ and $W^{(t)}$ are in $\mathfrak{g}$,
- the *global state* can be defined in $\mathfrak{g}$!



[18] Kong et al., 2022; Lezcano-Casado and Martınez-Rubio, 2019.

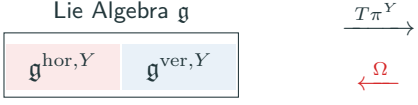
Step 2 — Homogeneous Spaces: the Projection $\pi : G \to \mathcal{M}$

- the projection $\pi : G \to \mathcal{M}$ is defined as $g \mapsto \pi(g) := g \cdot E$,
- π is surjective but not injective; $H_Y = \pi^{-1}(Y)$ contains many elements,
- a section $\lambda : \mathcal{M} \to G$ is a right-inverse, mapping $Y \in \mathcal{M}$ to a *single* $\lambda(Y)$ in the fiber above it,
- and one $\lambda(Y)$ moves the whole fibre: $H_Y = \lambda(Y)H$, so $\lambda(Y)^{-1}$ carries H_Y onto the fixed H — the same move the next steps make for $\mathfrak{g}^{\mathrm{hor},Y}$.



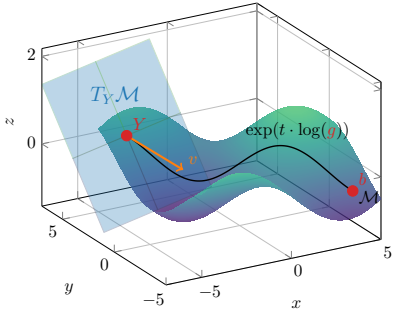
The Lie Algebra Decomposition $\mathfrak{g} = \mathfrak{g}^{\mathrm{ver},Y} \oplus \mathfrak{g}^{\mathrm{hor},Y}$

- the map $\mathfrak{g} \to \mathfrak{g} \cdot Y = T_Y\mathcal{M}$ is surjective[19],
- vertical subspace $\mathfrak{g}^{\mathrm{ver},Y} \implies$ kernel of this map,
- horizontal subspace $\mathfrak{g}^{\mathrm{hor},Y} \implies$ orthogonal complement of $\mathfrak{g}^{\mathrm{ver},Y}$,
- decomposition: $\mathfrak{g} = \mathfrak{g}^{\mathrm{ver},Y} \oplus \mathfrak{g}^{\mathrm{hor},Y}$.



- the isomorphism $\Omega : T_Y\mathcal{M} \cong \mathfrak{g}^{\mathrm{hor},Y}$ takes $V_Y \in T_Y\mathcal{M}$ and returns the unique element in $\mathfrak{g}^{\mathrm{hor},Y}$ that generates V_Y.
- for the Stiefel manifold:

$$\Omega(V_Y) = (\mathbb{I} - \tfrac{1}{2}YY^T)V_Y Y^T - YV_Y^T(\mathbb{I} - \tfrac{1}{2}YY^T)$$



[19] Edelman et al., 1998.

Global Sections: Mapping $\mathfrak{g}^{\mathrm{hor},Y}$ to the Fixed $\mathfrak{g}^{\mathrm{hor}}$

1. Definition

- $\lambda : \mathcal{M} \to G$ provides a right-inverse to $\pi : G \to \mathcal{M}$:

$$\pi(\lambda(Y)) \equiv \lambda(Y)E = Y \quad \forall Y \in \mathcal{M}.$$

2. Algorithmic Purpose: Enabling the Lift

- with $\lambda(Y)$ we can find the isomorphism between $\mathfrak{g}^{\mathrm{hor},Y}$ and $\mathfrak{g}^{\mathrm{hor}}$:

$$\lambda(Y)^{-1}\Omega(V_Y)\lambda(Y) \in \mathfrak{g}^{\mathrm{hor}}.$$

- this allows all Adam updates in the fixed $\mathfrak{g}^{\mathrm{hor}}$.

3. Construction for $St(n, N)$[20]

- Find a matrix $\Lambda \in SO(N)$ whose first n columns coincide with $Y \in St(n, N)$.

Method: Orthonormal Basis Completion

- Sample a random matrix $A \in \mathbb{R}^{N\times(N-n)}$.
- Compute: $A_\perp = A - YY^TA$.
- Perform a QR decomposition on $A_\perp$.
- The section is given by $\lambda(Y) = [Y|Q]$.

[20] While the choice of section $\lambda(Y)$ is not unique, the resulting geodesic on the manifold is independent of this choice.

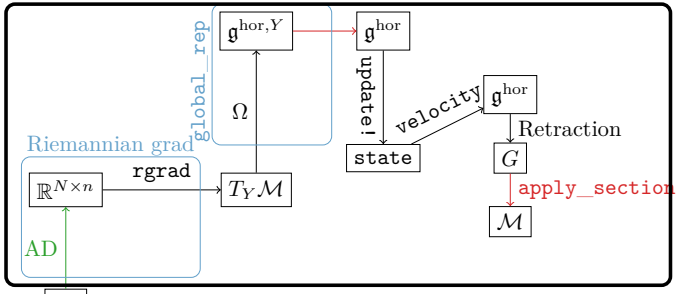
The Manifold Adam Framework

Generalize Adam by creating a path $T_Y\mathcal{M} \to \mathfrak{g}^{\mathrm{hor}}$[21].

four steps in an optimization step:

1. **Riemannian Gradient (`rgrad`):** Map the Euclidean gradient ∇L to $T_Y\mathcal{M}$.
2. **Lift to Global Tangent Space:** `global_rep` maps from $T_Y\mathcal{M}$ to the global $\mathfrak{g}^{\mathrm{hor}}$, using the section $\lambda(Y)$.

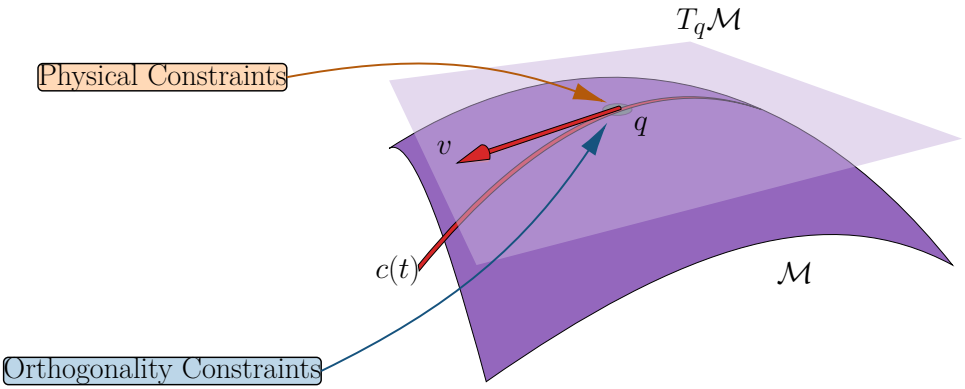
Key Advantage: perform Adam's vector-space operations in $\mathfrak{g}^{\mathrm{hor}}$ without requiring projections.



3. **Standard Adam Update:** The core Adam logic (`update!` and `velocity`) is performed on $\mathcal{B}^{\mathrm{state}} \in \mathfrak{g}^{\mathrm{hor}}$.
4. **Retraction to Manifold:** The velocity vector in $\mathfrak{g}^{\mathrm{hor}}$ is used to produce the next iterate $Y^{(t+1)} \in \mathcal{M}$.

[21] B., 2025.

Two Independent Reasons to Be on a Manifold



- **physical constraints** — the architecture leaves you no choice: a symplectic autoencoder cannot be trained in $\mathbb{R}^{N\times n}$ (B. & Kraus, 2023);
- **orthogonality constraints** — nothing forces you, but training is better, and sometimes only then possible (Kong et al., 2022; Lezcano-Casado & Martınez-Rubio, 2019; Li et al., 2020).

The next slide is the second one: an *ordinary* machine-learning task.

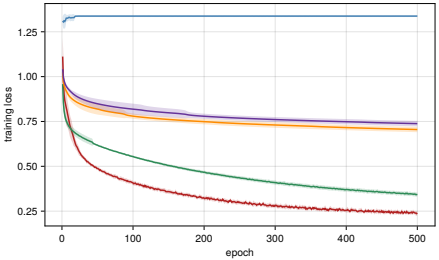
Numerical Validation: a Vision Transformer on the Stiefel Manifold

Setup: vision transformer, 28×28 images in 16 patches of 7×7; W^Q, W^K, W^V on $St(7, 49)$[22]; 10 fixed paired seeds $\times$ 500 epochs, Cayley retraction.

Key findings

1. standard Adam in $\mathbb{R}^{N \times n}$ fails to train — $10.7 \pm 0.7\%$ test accuracy, i.e. chance,
2. the second moment has to be coordinate-wise: against the scalar-moment Riemannian Adam of Li et al., 2020, otherwise identical: $86.0 \pm 0.6\%$ vs. $76.2 \pm 0.8\%$ on MNIST, $78.7 \pm 0.5\%$ vs. $75.4 \pm 0.6\%$ on Fashion-MNIST (appendix).

$\implies$ the manifold approach *simplifies* training; no need for layer normalization, dropout, clipping ...

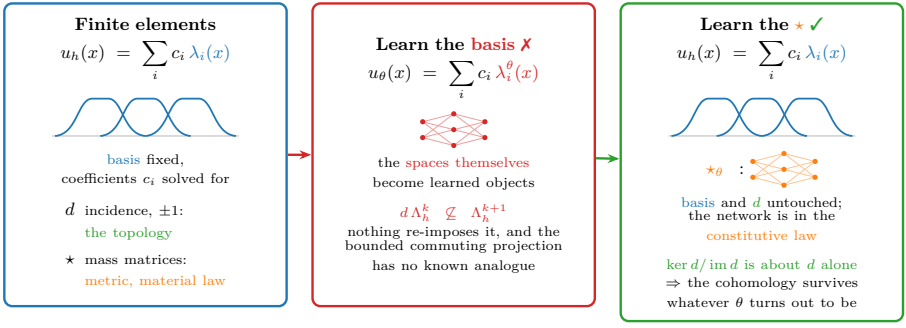


Training loss on MNIST, ten seeds.

- Geometric Adam
- Scalar-moment Adam
- Riemannian momentum
- Riemannian gradient
- Standard Adam

[22] Kong et al., 2022.

What This Looks Like for Neural Networks



- **the DEC split** (Desbrun et al., 2005): a coboundary d that is pure incidence, ± 1 and topological; and a Hodge star $\star$, the mass matrices, where metric and material law live,
- $d \circ d = 0$ and $\ker d / \operatorname{im} d$ are about d alone $\Rightarrow$ fix d, learn $\star$ — which is the right-hand panel, and what all three constructions do: the splines are not replaced (Kinch et al., 2025; Longo et al., 2023; Trask et al., 2022),
- missing: the middle panel. For a learned *basis* nothing re-imposes $d\,\Lambda_h^k \subseteq \Lambda_h^{k+1}$, and the bounded commuting projection has no known analogue.

Open Questions

What is Not Known

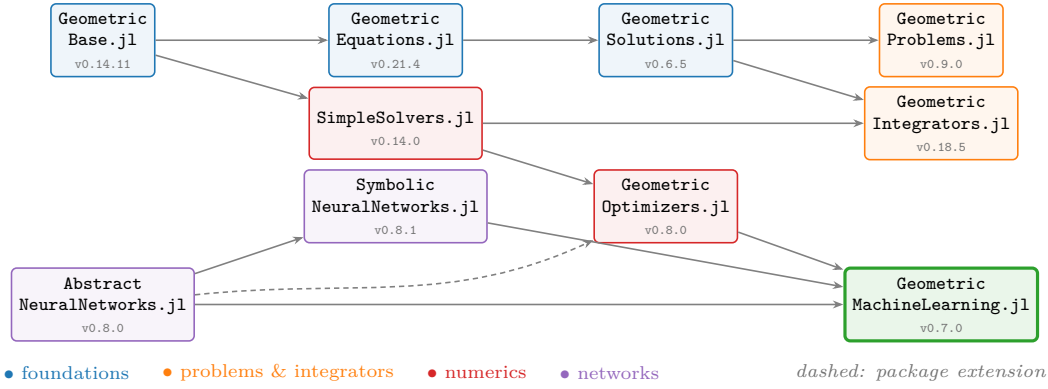
On the architecture side

- no backward error analysis for learned flow maps: we know a SympNet is symplectic — we do not know *which* modified Hamiltonian it integrates, so the Part I argument does not close.
- the FEEC side is wide open: what is the analogue of a *commuting projection* for a neural network?
- dissipative, forced and stochastic systems: begun (B. & Tyranowski, 2026).

On the optimization side

- a study of holonomy and lift-invariance is essentially absent — we have the construction, not the guarantees.
- which retraction? Geodesic and Cayley give different iterates. Does the choice change the minimizer that is found, or only the cost of reaching it?
- scaling: Ω, $\lambda(Y)$ and $\exp$ all cost more than a matrix multiply.
- can the *optimizer* itself be a geometric integrator of a gradient flow?

Software



Why Julia? the package manager is part of the language, so an environment is a `Project.toml` and a run is reproducible; just-in-time compilation, so the readable version is the fast one and there is no two-language problem; `Documenter.jl` builds the manual from the docstrings and runs its examples as tests; automatic differentiation composes with the custom manifold array types, so the gradient arrives in a form `lift` and `retraction` can consume; and the code mirrors the notation — $\lambda, \Omega, \mathfrak{g}, \nabla$ are legal identifiers, which makes it readable by a mathematician and by a language model.

Each package depends only on those to its left. Eleven more registered packages are not shown. github.com/JuliaGNI, registered versions of September 2026.

Summary

- **Part I.** In scientific computing we long ago stopped asking only "is the method accurate?" and started asking "*what does it preserve?*" — because approximability and consistency do not imply stability.
- **Part II.** A network used as a flow map is a numerical method, so the same question applies to it. Encoding the structure in the architecture buys long-time reliability and data efficiency; it costs expressivity, and the approximation theory is not there yet.
- **Part III.** Doing that puts the weights on a manifold, and the optimizers deep learning actually uses were not built for manifolds. Homogeneous-space structure — a global $\mathfrak{g}^{\mathrm{hor}}$ reached through a section — gives Adam somewhere to keep its state.

And a closing remark: none of Part III is physics. Orthonormal weights, bounded spectra and geometric optimizers are useful for **ordinary machine learning** too — the MNIST result was not a detour.

References i

Absil, P.-A., Mahony, R., & Sepulchre, R. (2008). ***Optimization algorithms on matrix manifolds.*** Princeton University Press.

Arnold, D. N., Falk, R. S., & Winther, R. (2006). **Finite element exterior calculus, homological techniques, and applications.** *Acta numerica*, *15*, 1–155.

Bajārs, J. (2023). **Locally-symplectic neural networks for learning volume-preserving dynamics.** *Journal of Computational Physics*, *476*, 111911.

Boffi, D. (2010). **Finite element approximation of eigenvalue problems.** *Acta Numerica*, *19*, 1–120.

Bossavit, A. (1990). **Solving Maxwell equations in a closed cavity, and the question of 'spurious modes'.** *IEEE Transactions on Magnetics*, *26*(2), 702–705.

Boumal, N. (2023). ***An introduction to optimization on smooth manifolds.*** Cambridge University Press.

Brantner, B. (2025). **Generalizing adam to manifolds for efficiently training transformers.** *arXiv preprint arXiv:2305.16901v4 (submitted to SN Computer Science)*.

References ii

Brantner, B., & Kraus, M. (2023). **Symplectic autoencoders for model reduction of hamiltonian systems.** *arXiv preprint arXiv:2312.10004*.

Brantner, B., Kraus, M., Li, Z., & de Romemont, G. (2025). **Volume-preserving transformers for learning time series data with structure.** *ESAIM: Proceedings and Surveys*, *81*, 123–144.

Brantner, B., & Tyranowski, T. M. (2026). **Learning deterministic and stochastic forced hamiltonian systems.** *arXiv preprint arXiv:2608.19688*.

Bronstein, M. M., Bruna, J., Cohen, T., & Veličković, P. (2021). **Geometric deep learning: Grids, groups, graphs, geodesics, and gauges.** *arXiv preprint arXiv:2104.13478*.

Buchfink, P., Glas, S., & Haasdonk, B. (2023). **Symplectic model reduction of hamiltonian systems on nonlinear manifolds and approximation with weakly symplectic autoencoder.** *SIAM Journal on Scientific Computing*, *45*(2), A289–A311.

Cartan, É. (1930). **La théorie des groupes finis et continus et l'analysis situs.**

Cohen, T., & Welling, M. (2016). **Group equivariant convolutional networks.** *International Conference on Machine Learning*, 2990–2999.

References iii

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., & Ho, S. (2020). **Lagrangian neural networks.** *arXiv preprint arXiv:2003.04630; published at ICLR 2020 Deep Differential Equations Workshop.*

Desbrun, M., Hirani, A. N., Leok, M., & Marsden, J. E. (2005). **Discrete exterior calculus.** *arXiv preprint math/0508341.*

Dierkes, E., Offen, C., Ober-Blöbaum, S., & Flaßkamp, K. (2023). **Hamiltonian neural networks with automatic symmetry detection.** *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *33*(6), 063115.

Edelman, A., Arias, T. A., & Smith, S. T. (1998). **The geometry of algorithms with orthogonality constraints.** *SIAM journal on Matrix Analysis and Applications*, *20*(2), 303–353.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). ***Deep learning*.** MIT press.

Greydanus, S., Dzamba, M., & Yosinski, J. (2019). **Hamiltonian neural networks.** *Advances in neural information processing systems*, *32*.

References iv

Güçlü, Y., Hadjout, S., & Ratnani, A. (2022). **PSYDAC: A high-performance IGA library in Python.** *8th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2022)*. https://doi.org/10.23967/eccomas.2022.227

Hairer, E. (2014). **Challenges in geometric numerical integration.** In *Trends in contemporary mathematics* (pp. 125–135, Vol. 8). Springer.

Hairer, E., Lubich, C., & Wanner, G. (2003). **Geometric numerical integration illustrated by the störmer–verlet method.** *Acta Numerica*, *12*, 399–450.

Hairer, E., Lubich, C., & Wanner, G. (2006). ***Geometric numerical integration: Structure-preserving algorithms for ordinary differential equations.*** Springer.

Horn, P., Ulibarrena, V. S., Koren, B., & Zwart, S. P. (2025). **A generalized framework of neural networks for hamiltonian systems.** *Journal of Computational Physics*, *521*, 113536.

Hornik, K., Stinchcombe, M., & White, H. (1989). **Multilayer feedforward networks are universal approximators.** *Neural Networks*, *2*(5), 359–366.

References v

Jin, P., Zhang, Z., Zhu, A., Tang, Y., & Karniadakis, G. E. (2020). **Sympnets: Intrinsic structure-preserving symplectic networks for identifying hamiltonian systems.** *Neural Networks*, *132*, 166–179.

Kinch, B., Shaffer, B., Armstrong, E., Meehan, M., Hewson, J., & Trask, N. (2025). **Structure-preserving digital twins via conditional neural Whitney forms.** *arXiv preprint arXiv:2508.06981*.

Kingma, D. P., & Ba, J. L. (2014). **Adam: A method for stochastic optimization [Published as a conference paper at ICLR 2015].** *arXiv preprint arXiv:1412.6980*.

Klein, F. (1872). ***Vergleichende betrachtungen über neuere geometrische forschungen: Programm zum eintritt in die philosophische facultät und den senat der k. friedrich-alexanders-universität zu erlangen.*** Deichert.

Kong, L., Wang, Y., & Tao, M. (2022). **Momentum stiefel optimizer, with applications to suitably-orthogonal attention, and optimal transport [Published as a conference paper at ICLR 2023].** *arXiv preprint arXiv:2205.14173*.

References vi

Lee, K., & Carlberg, K. T. (2020). **Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders.** *Journal of Computational Physics*, *404*, 108973.

Lezcano-Casado, M., & Martınez-Rubio, D. (2019). **Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group.** *Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.*, 3794–3803.

Li, J., Fuxin, L., & Todorovic, S. (2020). **Efficient Riemannian optimization on the Stiefel manifold via the Cayley transform [Published as a conference paper at ICLR 2020].** *arXiv preprint arXiv:2002.01113*.

Lishkova, Y., Scherer, P., Ridderbusch, S., Jamnik, M., Liò, P., Ober-Blöbaum, S., & Offen, C. (2023). **Discrete Lagrangian neural networks with automatic symmetry discovery.** *IFAC-PapersOnLine*, *56*(2), 3203–3210.

Longo, M., Opschoor, J. A. A., Disch, N., Schwab, C., & Zech, J. (2023). **De Rham compatible deep neural network FEM.** *Neural Networks*.

References vii

Marsden, J. E., & West, M. (2001). **Discrete mechanics and variational integrators.** *Acta Numerica*, *10*, 357–514.

Nomizu, K. (1954). **Invariant affine connections on homogeneous spaces.** *American Journal of Mathematics*, *76*(1), 33–65.

Offen, C., & Ober-Blöbaum, S. (2022). **Symplectic integration of learned Hamiltonian systems.** *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *32*(1), 013122.

O'neill, B. (1983). ***Semi-riemannian geometry with applications to relativity.*** Academic press.

Peng, L., & Mohseni, K. (2016). **Symplectic model reduction of hamiltonian systems.** *SIAM Journal on Scientific Computing*, *38*(1), A1–A27.

Qin, H. (2024). **Advanced fuel fusion, phase space engineering, and structure-preserving geometric algorithms.** *Physics of Plasmas*, *31*(5).

Trask, N., Huang, A., & Hu, X. (2022). **Enforcing exact physics in scientific machine learning: A data-driven exterior calculus on graphs.** *Journal of Computational Physics*, *456*, 110969.

References viii

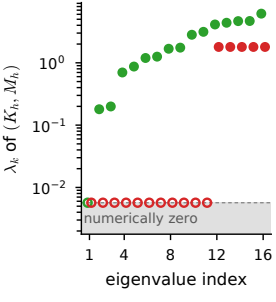
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). **Attention is all you need.** *Advances in neural information processing systems*, *30*.

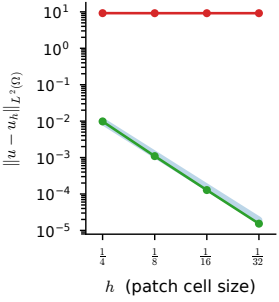
Appendix

Conformity, Separately: the Poisson Solve on the Pretzel

- $-\Delta u = f$ on the same pretzel, one scalar space, discretized twice — same spaces, same degree, same solver — differing only in whether the patches are tied together across their 13 interfaces,
- the conformity half of the story: the naive variant is not under-solved and not unstable, it converges to a different problem.





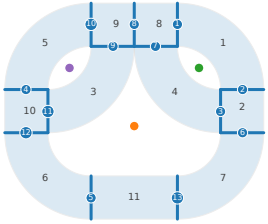


tied across the interfaces — assembled patch by patch — degree 2, 4 to 32 cells per patch.

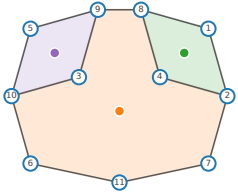
left u_h, finest grid · *middle* the 16 smallest $\lambda_k(K_h, M_h)$: **one** zero mode v. **eleven** · *right* observed order **3.11** v. **0.00**.

Backup: What the 13 Interfaces Are, and Where $13 - 11 + 1$ Comes From

the domain: 11 patches, glued along 13 interfaces



the patch graph: a node per patch, an edge per interface



$b_1 = 13 - 11 + 1 = 3$

- each patch is the image of a logical square under its own spline mapping; an interface is one edge shared by exactly two patches — 11 of the first, 13 of the second, and *tying* the patches happens there,
- both are contractible, so the graph on the right carries the topology of the domain, and for a graph the Betti numbers are counted rather than approximated: $b_0 = \#\text{components} = 1$, $b_1 = \#\text{edges} - \#\text{nodes} + \#\text{components} = 13 - 11 + 1 = 3$ — the three shaded faces, one per hole — the dots mark them in both panels.

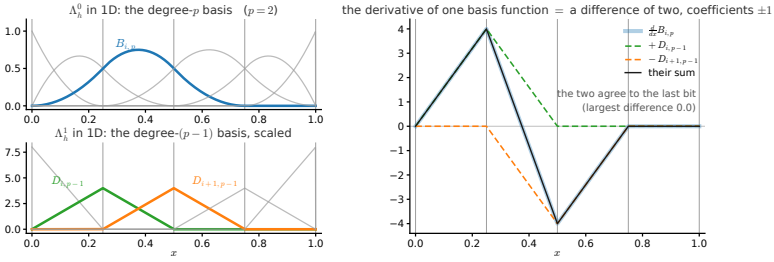
Backup: Tied *Compatibly*, and Where the 432 Come From

- it is *not* "more continuity is better": every rung has to be tied, *compatibly*,
- tie Λ_h^0, leave Λ_h^1 broken $\Rightarrow \Lambda_h^1 \not\subset H(\mathrm{curl}, \Omega)$: still a complex, not a *sub*complex. $\ker \mathrm{curl}_h$ is the *patchwise* gradients, $\mathrm{im}\, \mathrm{grad}_h$ only the *tied* ones — one spurious field per continuity constraint the tying imposed:

$$\dim \mathcal{H}_h^1 = 13(n+2) - 10 = 68,\ 120,\ 224,\ 432 \quad \text{at } n = 4,\ 8,\ 16,\ 32 \text{ cells per patch,}$$

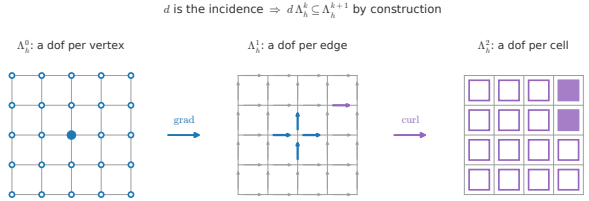
- tie nothing and the three holes disappear instead. Here not even the bottom rung is conforming: a function of the broken Λ_h^0 is a separate spline per patch and jumps across every interface, so $\Lambda_h^0 \not\subset H^1(\Omega)$ — no continuity, let alone a matching tangential trace one rung up. grad_h then has nothing to cancel and $\ker \mathrm{curl}_h$ nothing to contain: b_1 collapses to 0 and b_0 counts the patches, 11. *The two failures point opposite ways*, so "tie more" is not the axis,
- the symptom, classically: spectra that "include spurious eigenvalues that converge to a point outside the true spectrum as the mesh is refined" (Arnold et al., 2006),
- $\dim \mathcal{H}_h^0$: 1 tied, 1 with Λ_h^0 only, 11 with nothing tied — one per patch.

Backup: Why the Ladder Drops Degree — the Spline Fact



- in 1D, $\frac{d}{dx}B_{i,p} = D_{i,p-1} - D_{i+1,p-1}$: the derivative of one degree-p basis function is a difference of two degree-$(p-1)$ ones on the same cells, coefficients ± 1 — no mesh size, no geometry, no quadrature,
- in 2D, once per direction, because differentiating lowers the degree only in the direction differentiated — and that is the ladder: $S^{p,p} \to S^{p-1,p} \times S^{p,p-1} \to S^{p-1,p-1}$,
- **what it costs:** Λ_h^1 is one order lower in L^2 ($\mathcal{O}(h^p)$ v. $\mathcal{O}(h^{p+1})$) — and is the smaller space, 660 v. 792 at $n = 4$, $p = 2$. *One order of approximation for an exact integer*, and only the integer cannot be had by refining.

Backup: the FEEC Construction, on a Patch and Across an Interface



top d is incidence, so $d\,\Lambda_h^k \subseteq \Lambda_h^{k+1}$ holds by construction and the matrix of d is written down, not integrated · *bottom* each rung is tied by the trace that rung needs — tie all three alike and the complex breaks, and so does leaving a rung untied · drawn at lowest order, where dof $\leftrightarrow$ vertex / edge / cell is exact; the study uses $p = 2$.

Why it Works: Backward Error Analysis

- a symplectic ψ_h is — up to exponentially small terms — the *exact* flow of a modified Hamiltonian[23]

$$H_h = H + h^p H_{p+1} + h^{p+1} H_{p+2} + \cdots$$

- H_h is conserved exactly $\implies$

$$|H(z_n) - H(z_0)| = \mathcal{O}(h^p) \quad \text{for } t \lesssim e^{c/h}.$$

- the method does not solve *our* problem approximately; it solves a *nearby physical problem* exactly.

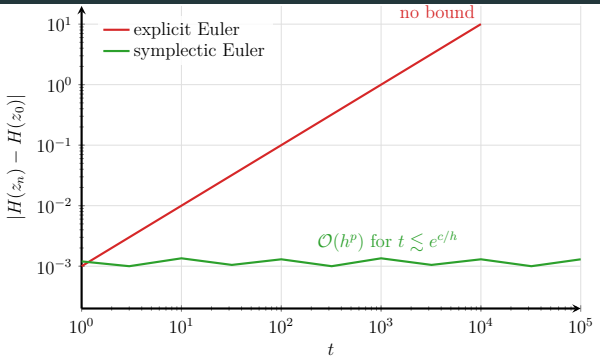


Figure 6: Energy error over long times. *Schematic.*

A non-symplectic method has no such H_h — nothing bounds the drift.

[23] Hairer et al., 2006.

Discretise the Action, not the Equations

- **variational integrators**[24]: approximate the action rather than the vector field,

$$S = \int_0^T L(q, \dot{q})\, dt \;\rightsquigarrow\; S_d = \sum_k L_d(q_k, q_{k+1}),$$

- stationarity of S_d gives the discrete Euler–Lagrange equations,
- the resulting method is symplectic by construction — symplecticity is not imposed, it is inherited,
- a *discrete* symmetry of L_d gives an *exactly* conserved discrete momentum (discrete Noether theorem).

The pattern. Preserve the object the physics is *stated in* — the symplectic form, the action, the symmetry — and the conservation laws follow for free.

- this is a whole discipline: RK / partitioned RK, splitting & composition, VPRK, SPARK, DVI, CGVI / DGVI, Hamilton–Pontryagin, …
- see `GeometricIntegrators.jl` for a library of them.

[24] Marsden, J. E., & West, M. (2001). **Discrete mechanics and variational integrators.** *Acta Numerica*, *10*, 357–514.

The Data-Driven Paradigm

Machine learning is the "ability to acquire [...] knowledge, by extracting patterns from raw data."[25]

- this is used for:
  - image classification,
  - natural language processing,
  - making medical diagnoses,
  - ...
- when dealing with differential equations: purely data-driven approaches may not work well! $\implies$ use traditional numerics as inspiration.

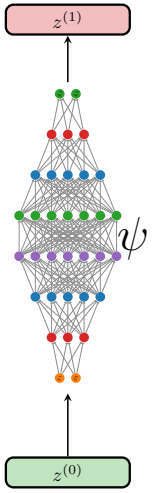


Figure 7: ψ as numerical flow.

[25] Goodfellow et al., 2016.

Lagrangian Neural Networks

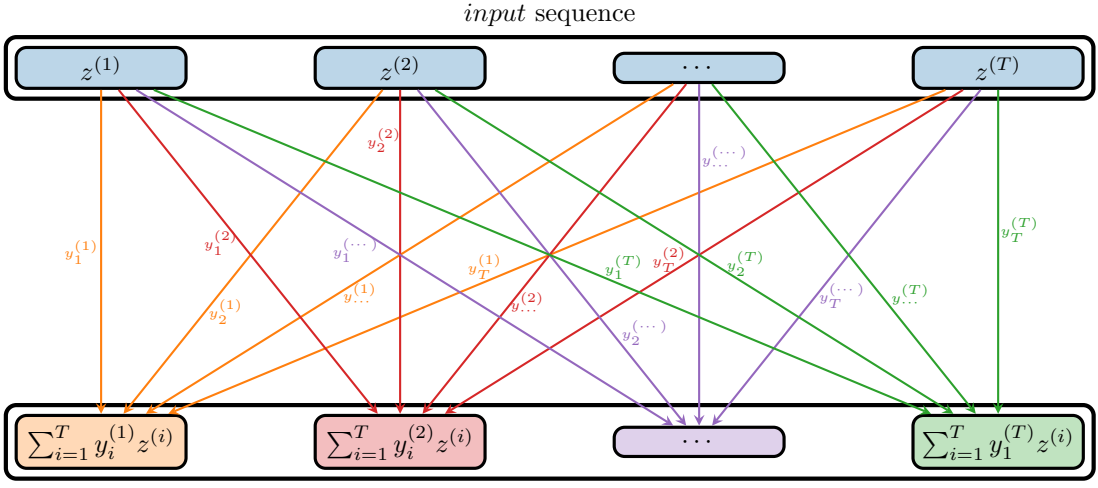
- *discovery* of a Lagrangian from data[26],
- use

$$\ddot{q} = f_{\mathcal{L}}(q, \dot{q}) = \sum_t (\nabla_{\dot{q}} \nabla_{\dot{q}}^T \mathcal{L})^{-1} [\nabla_q \mathcal{L} - (\nabla_q \nabla_{\dot{q}}^T \mathcal{L}) \dot{q}]$$

  to compute an error based on data $x_t = (q^{(t)}, \dot{q}^{(t)})$,
- $\mathcal{L}$ is modeled with a neural network,
- integration can be performed with a variational integrator — the natural partner, by Part I.

[26] Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., & Ho, S. (2020). **Lagrangian neural networks.** *arXiv preprint arXiv:2003.04630; published at ICLR 2020 Deep Differential Equations Workshop.*

Structure-Preserving Attention ≡ Learned Orthonormal Reweighting



Cayley $\implies (y_\bullet^{(i)})^T y_\bullet^{(j)} = \delta_{ij}$

Symplectic Attention

- two different ways of computing symplectic attention, based on:
  1. $\Sigma(Z) = \log(1 + \exp(\sum_{m,n} C_{mn}))$ (leading to the matrix softmax),
  2. $\Sigma(Z) = \sum_n \log(1 + \exp(\sum_m C_{mn}))$ (leading to the vector softmax),
- in both cases we update with a *multistep version of symplectic Euler*:
  - $Q \mapsto Q + \nabla\Sigma_p(P)$,
  - $P \mapsto P - \nabla\Sigma_q(Q)$.

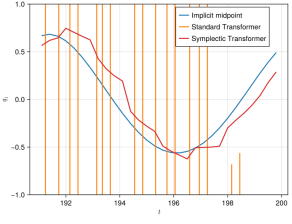
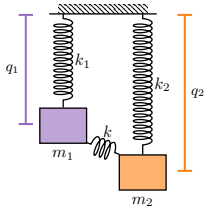
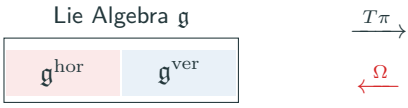


Figure 8: Evolution of q_1.



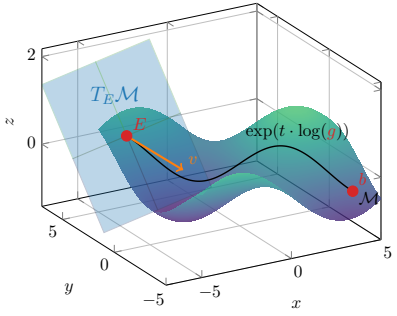
The Lie Algebra Decomposition at E

- the map $T\pi : \mathfrak{g} \to \mathfrak{g} \cdot E = T_E\mathcal{M}$ is surjective,
- vertical subspace $\mathfrak{g}^{\mathrm{ver}} \implies$ kernel of this map,
- horizontal subspace $\mathfrak{g}^{\mathrm{hor}} \implies$ orthogonal complement of $\mathfrak{g}^{\mathrm{ver}}$,
- decomposition: $\mathfrak{g} = \mathfrak{g}^{\mathrm{ver}} \oplus \mathfrak{g}^{\mathrm{hor}}$.

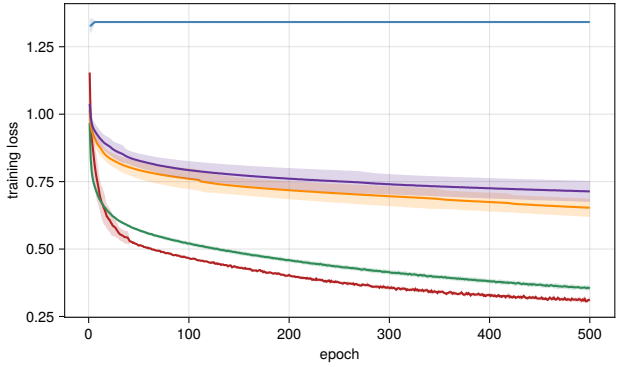


- the isomorphism $\Omega : T_E\mathcal{M} \cong \mathfrak{g}^{\mathrm{hor}}$,
- for the Stiefel manifold:

$$\Omega(V) = (\mathbb{I} - \tfrac{1}{2}EE^T)VE^T - EV^T(\mathbb{I} - \tfrac{1}{2}EE^T)$$



The Same Comparison on Fashion-MNIST



Training loss on Fashion-MNIST, Cayley retraction, ten fixed paired seeds; solid lines are means, bands one sample standard deviation.

What the Geometry Costs: Timings for Geometric Adam

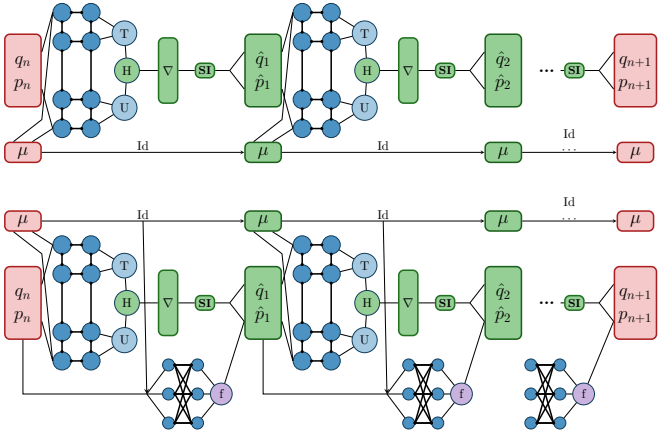
The manifold buys two extra operations per step: mapping the Euclidean gradient into $\mathfrak{g}^{\mathrm{hor}}$, and applying the retraction. Both are timed separately below; neither is the bottleneck, because automatic differentiation through 16 transformer blocks at batch size 2048 dwarfs them.

	time per step (ms)			geometric	time per	peak dev.
configuration	grad. (AD)	direction	retraction	share (%)	epoch (s)	mem. (GiB)
geometric Adam	118.4	**4.85**	**20.83**	**17.8**	7.67	21.8
scalar-moment Adam	115.8	5.33	36.15	25.8	5.63	22.2
Riemannian momentum	121.5	4.33	41.39	25.6	7.91	23.5
Riemannian gradient	131.2	5.37	32.95	22.5	8.73	23.4
standard Adam (does not train)	119.7	9.26	0.30	7.4	7.21	22.6

MNIST, Cayley retraction, NVIDIA GeForce RTX 4090, `Float32`, batch size 2048; Table 4 of B. (2025). *Geometric share* is direction plus retraction as a fraction of the instrumented step. Optimizer state: 4.34 MiB against 1.77 MiB for standard Adam.

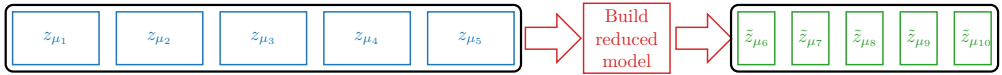
The geometry is about a fifth of the step and $\sim 10\%$ of the run; the optimizer that stays in $\mathbb{R}^{N\times n}$ is not cheaper by enough to matter, and it does not train.

Adjustment to Other Settings



parametrised and forced GHNNs (B. & Tyranowski, 2026)

Data-Driven Reduced Order Modeling



For a reduced order modeling framework we need two sets of mappings:

1. for the offline phase we build an encoder $\Psi^{\text{enc}} : \text{FOM} \to \text{ROM}$ and a decoder $\Psi^{\text{dec}} : \text{ROM} \to \text{FOM}$,
2. for the online phase we need to build a propagator $\psi : \text{ROM} \to \text{ROM}$.

