Equivariant Eikonal Neural Networks

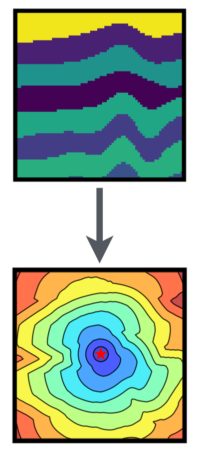
Grid-free, scalable travel-time prediction on homogeneous spaces

Alejandro García-Castellanos, David R. Wessels, Nicky J. van den Berg, Remco Duits, Daniël M. Pelt, Erik J. Bekkers

The Eikonal Equation

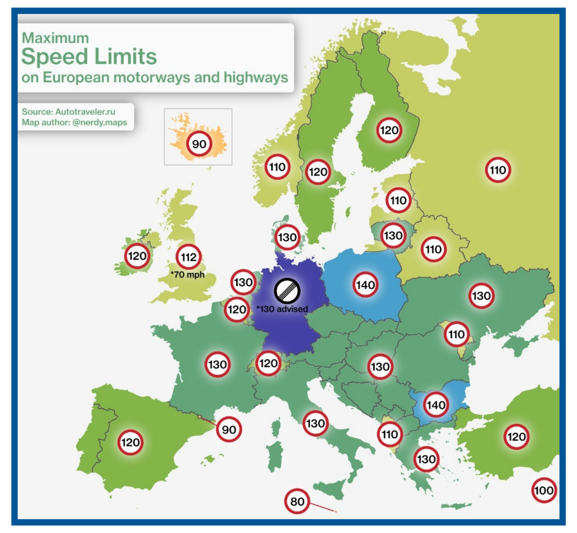
On a Riemannian manifold $(\mathcal{M}, \mathcal{G})$, the two-point Riemannian Eikonal equation with respect to a velocity field $v : \mathcal{M} \to [v_{\min}, v_{\max}]$ (where $0 < v_{\min} \leq v_{\max} < \infty$) is:

$$\begin{cases} \|\operatorname{grad}_s T(s,r)\|_{\mathcal{G}} = v(s)^{-1}, \\ \|\operatorname{grad}_r T(s,r)\|_{\mathcal{G}} = v(r)^{-1}, \\ T(s,r) = T(r,s), \quad T(s,s) = 0, \end{cases}$$



where grad_s and grad_r denote the Riemannian gradients with respect to the source $s \in \mathcal{M}$ and the receiver $r \in \mathcal{M}$, respectively

The solution $T : \mathcal{M} \times \mathcal{M} \to \mathbb{R}_+$ corresponds to the travel-time function, i.e., the minimum time to travel from source to receiver given the velocity map

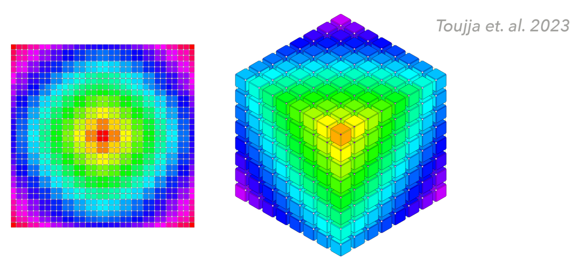


Ex: How long it would take to go from Spain to Estonia if we go at the maximum speed allowed at each moment?

Original Methods

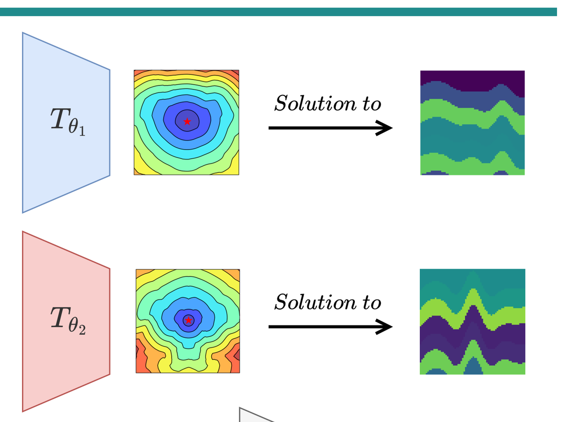
Fast Marching Method (FMM)

- Bad memory and computation scalability
- Requires discretization



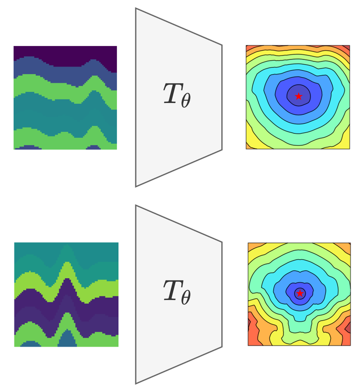
Physics-Informed Neural Networks (PINNs): encode eikonal equation into loss function

- Separate network for each solution → Memory inefficient
- Not sharing knowledge between solutions → Training inefficient

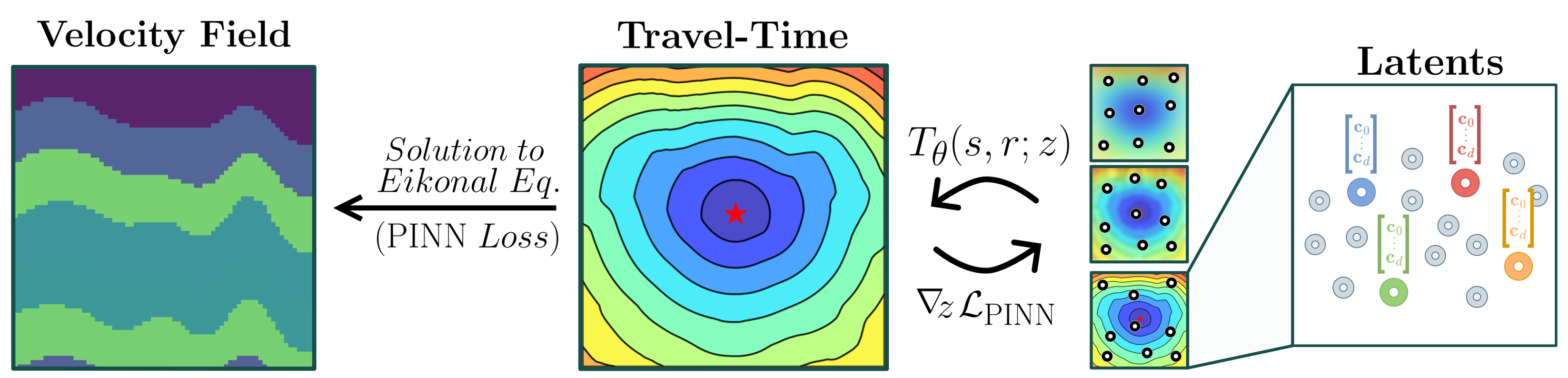


Neural Operators: share a common backbone across solutions and condition on velocities profiles

- Naive conditioning variables → Inefficient adaptability
- Current approaches are not fully grid free



We extend Equivariant Neural Fields



Instead of a global latent vector z, Equivariant Neural Fields are conditioned by cross-attention on a point cloud of pose-context tuples: $z := \{(g_i, \mathbf{c}_i)\}_{i=1}^N, g_i \in G, \mathbf{c}_i \in \mathbb{R}^d$

z is obtained by optimizing it using gradient descent on a PINN loss of the Eikonal Equation

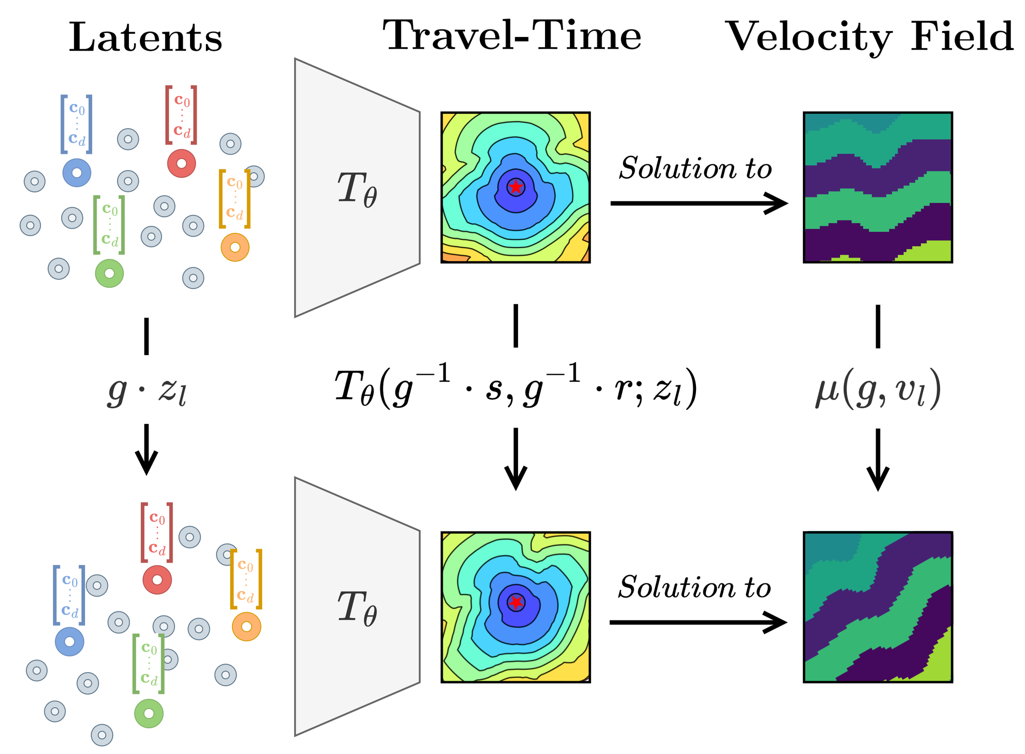
Steerability property

When you transform the conditioning variables the travel-time solution transforms predictably too!

Solving for one velocity field automatically gives you solutions for its entire orbit

We get steerability if and only if
$T_\theta(g \cdot s, g \cdot r; g \cdot z) = T_\theta(s, r; z)$

We need expressive invariants for our architecture



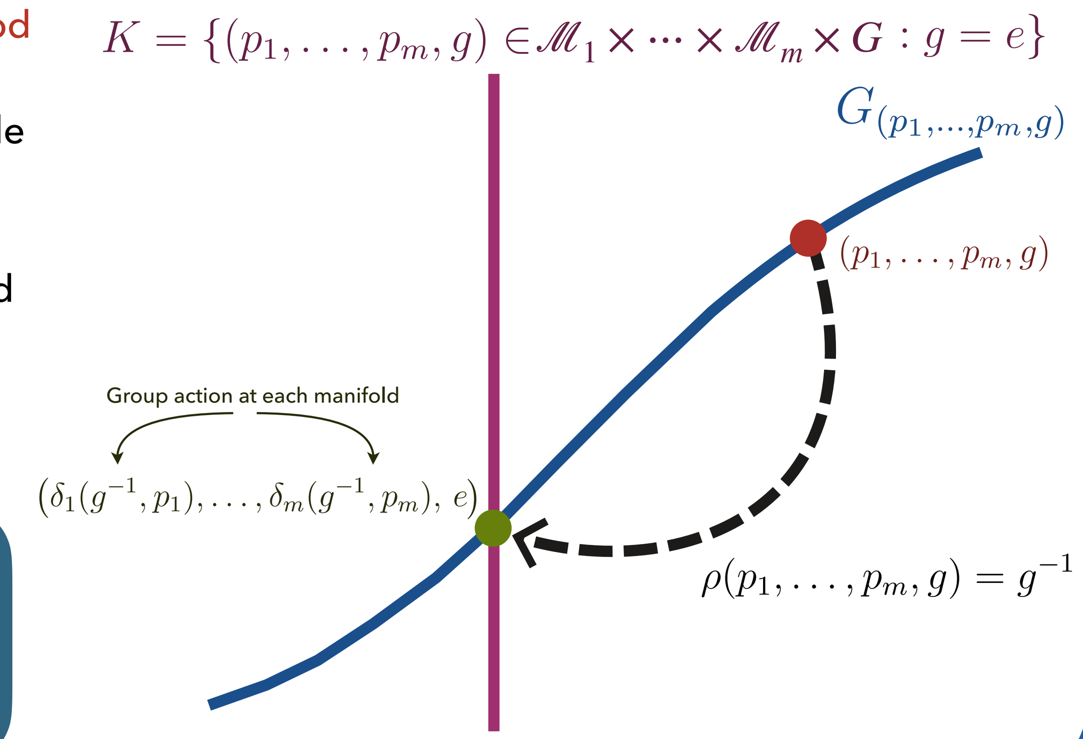
Computing Invariances

We extend the moving frame method to product of distinct manifolds by augmenting the space with learnable group elements

This makes non-free actions free and gives us a complete and maximally expressive set of independent invariants!

Unlocking equivariant neural fields on:

- Product manifolds
- Non-transitive actions

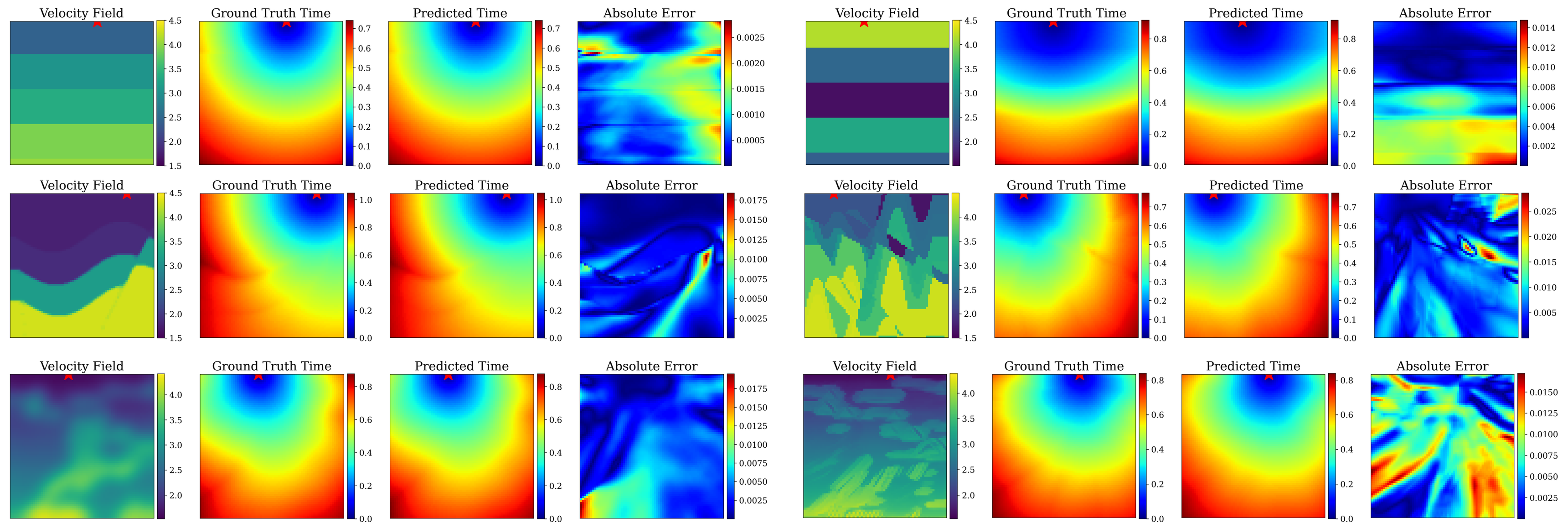


Experiments

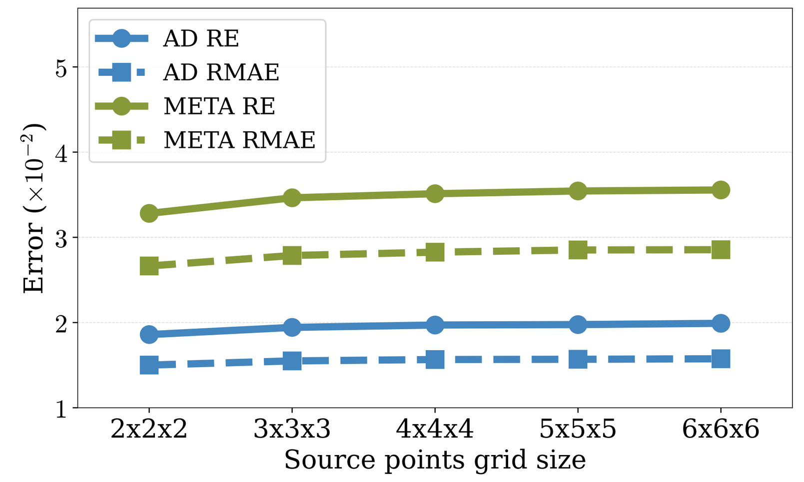
We validate our approach through applications in seismic travel-time modeling of 2D and 3D benchmark datasets

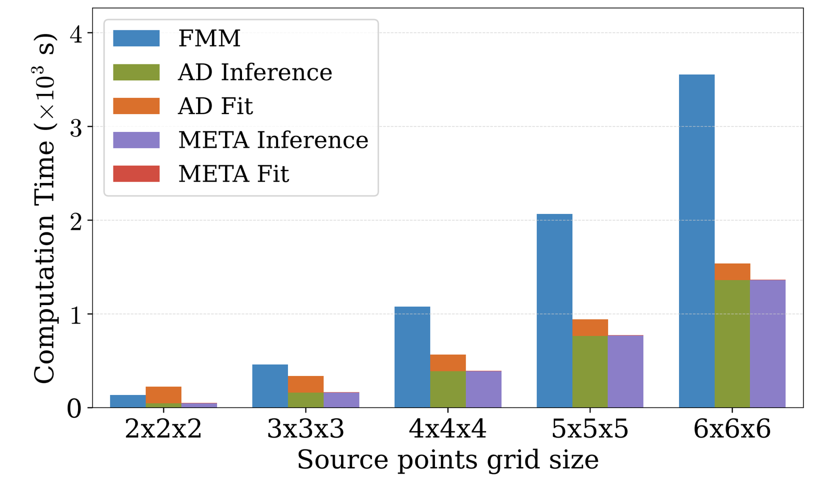
- E-NES with full autodecoding beats FC-DeepONet on 7/10 datasets.
- Even with meta-learning *(fast but less expressive)*, we still outperform SOTA in 4 challenging cases while achieving 100x speedup: ~1000s → <6s per velocity field

	FC-DeepONet		E-NES: Autodecoding (100 epochs)		E-NES: Autodecoding (convergence)		E-NES: Meta-learning	
Dataset	RE (↓)	Fitting (s)	RE (↓)	Fitting (s)	RE (↓)	Fitting (s)	RE (↓)	Fitting (s)
FlatVel-A	**0.00277**	~ 0.615	0.00952	223.31	0.00506	1120.25	0.01065	5.92
CurveVel-A	0.01878	~ 0.615	0.01348	222.72	**0.00955**	1009.67	0.02196	5.91
FlatFault-A	**0.00514**	~ 0.615	0.00857	222.61	0.00568	1014.45	0.01372	5.92
CurveFault-A	0.00963	~ 0.615	0.01108	222.89	**0.00820**	1123.90	0.02086	5.92
Style-A	0,03461	~ 0.615	0.01034	222.00	**0.00833**	1117.99	0.01317	5.92
FlatVel-B	**0.00711**	~ 0.615	0.01581	222.74	0.00860	1010.32	0.02274	5.91
CurveVel-B	0.03410	~ 0.615	0.03203	222.97	**0.02250**	1127.87	0.03583	5.90
FlatFault-B	0.04459	~ 0.615	0.01989	222.70	**0.01568**	1133.98	0.03058	5.93
CurveFault-B	0.07863	~ 0.615	0.02183	222.89	**0.01885**	893.84	0.03812	5.89
Style-B	0.03463	~ 0.615	0.01171	221.90	**0.01069**	896.06	0.01541	5.90

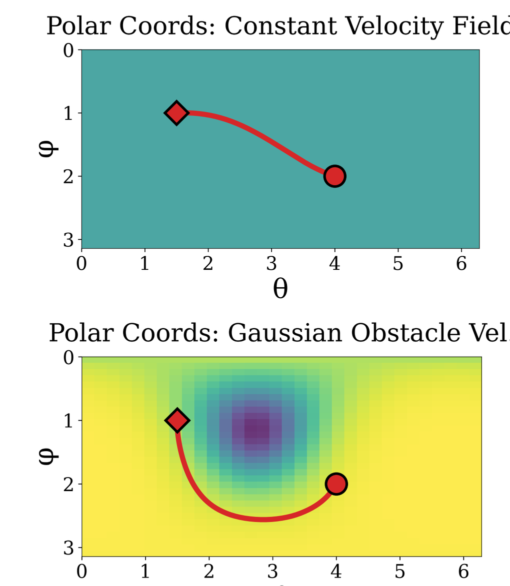


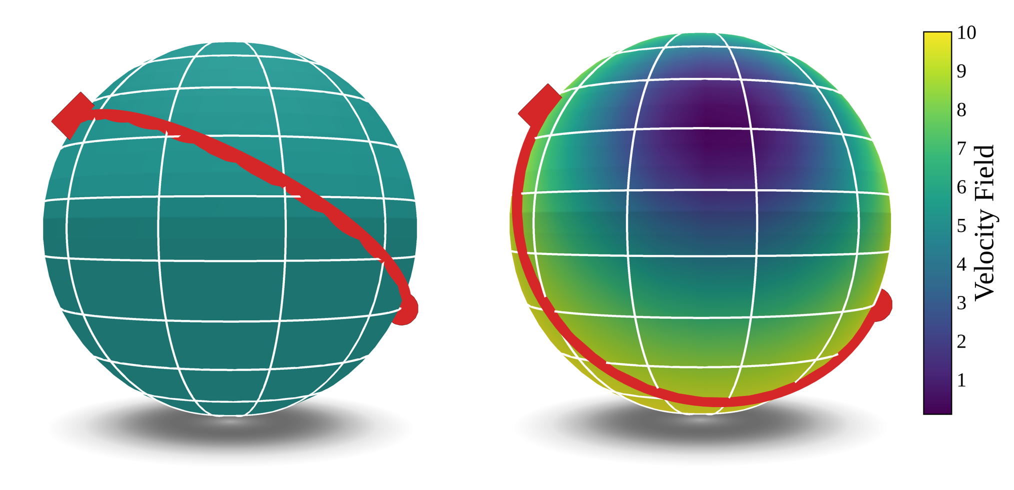
- We can extend to 3D while maintaining stable error metrics across increasing grid dimensions.
- In contrary to Fast Marching Method, we don't need for finer discretization as problems get larger thanks to E-NES continuous representation





We can perform geodesic path planning via Riemannian SGD over $\operatorname{grad}_s T(s, r)$, yielding optimal trajectories under configurations with and without obstacles





Constant Velocity Field   Gaussian Obstacle Velocity Field

Hybrid Intelligence
AMLAB Amsterdam Machine Learning Lab
University of Amsterdam
SCAN ME