Neural semi-Lagrangian schemes for high-dimensional advection-diffusion equations

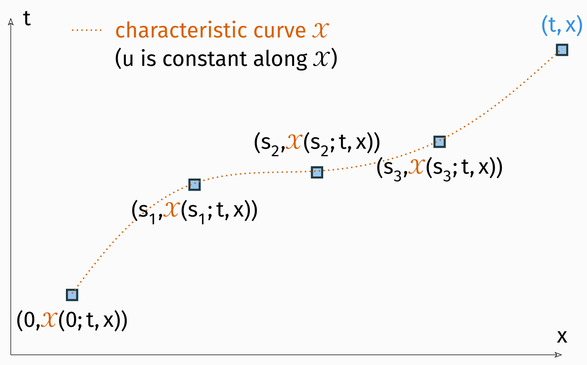
Emmanuel Franck[12], Victor Michel-Dansac[12], Laurent Navoret[21], Vincent Vigon[21]

[1]MACARON team at Inria Strasbourg, France [2]IRMA, Université de Strasbourg, France

1.1 - The advection equation and its characteristic curves

For $x \in \Omega \subset \mathbb{R}^d$ and $t \in [0, T]$, we consider the advection equation:

$$\partial_t u(t,x) + a(t,x) \cdot \nabla_x u(t,x) = 0.$$



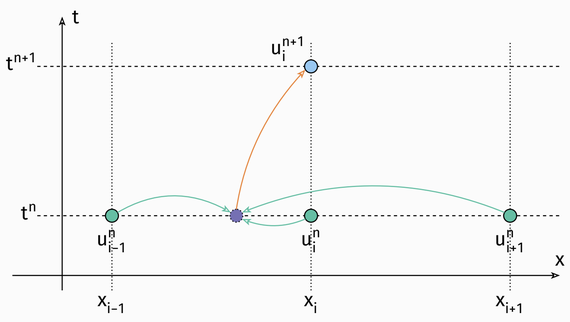
Its solution satisfies

$$\forall t \in [0,T], \quad \forall s \in [0,t], \quad \forall x \in \Omega, \quad u(t,x) = u(s, \mathcal{X}(s;t,x)),$$

where $\mathcal{X}$ solves the following backwards ODE for given t and x:

$$\begin{cases} \dfrac{d}{ds}\mathcal{X}(s;t,x) = a(s, \mathcal{X}(s;t,x)), & \forall s \in [0,t], \\ \mathcal{X}(t;t,x) = x. \end{cases}$$

1.2 - Classical semi-Lagrangian (SL) scheme [1]



classical SL scheme: update for one mesh point x_i from t^n to t^{n+1}

step 1: find ● from ● following the approximate characteristic curve

step 2: interpolate the value at ● from the values at ●

step 3: set u_i^{n+1} to the interpolated value

1.3 - High-dimensional parametric advection equations

We seek to treat high-dimensional advection equations:
- for dimensions $d \approx 6$ (common in e.g. plasma physics),
- with several physical parameters $\mu \in \mathbb{P} \subset \mathbb{R}^p$: u and a depend on t, x, and μ.

⇝ The problem is (1 + d + p)-dimensional!

The curse of dimensionality applies to classical (mesh-based) SL schemes:
- need for $O(N^d)$ grid points in space to represent u,
- need for many different simulations to cover parameter space.

⇝ For this reason, we turn to a neural method.

2.1 - Neural networks to solve unsteady PDEs

With $\mathcal{D}$ a spatial differential operator, consider an unsteady PDE of the form

$$\partial_t u + \mathcal{D}(u) = 0.$$

Classical explicit numerical methods usually discretize in time and space:

$$\forall i, \ \forall n \geq 0, \quad u_i^{n+1} = u_i^n - \Delta t \left(\mathcal{D}_\Delta(u^n)\right)_i.$$

Replicating this procedure using a neural network leads to time-sequential PINNs: for each n, approximate $x \mapsto u(t^n, x)$ by a neural network $x \mapsto u_{\theta^n}(x)$, and define

$$\forall n \geq 0, \quad \theta^{n+1} = \operatorname*{argmin}_\theta \int_\Omega \left| u_\theta(x) - u_{\theta^n}(x) + \Delta t\, \mathcal{D}(u_{\theta^n})(x) \right|^2 dx.$$

We wish to specialize this idea for advection equations.

2.2 - Neural semi-Lagrangian (NSL) scheme [2]

The exact solution u of the advection equation satisfies the characteristic property:

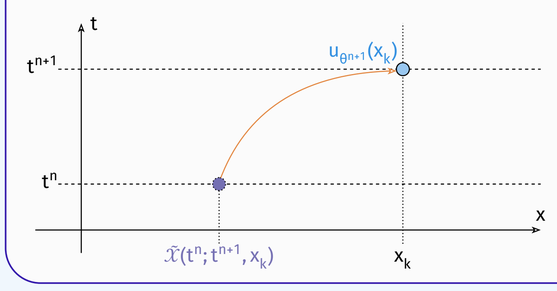
$$\forall x \in \Omega, \quad u\left(t^{n+1}, x\right) = u\left(t^n, \mathcal{X}(t^n; t^{n+1}, x)\right).$$

The approximate analogue of the characteristic property is then

$$\forall x \in \Omega, \quad u_{\theta^{n+1}}(x) \approx u_{\theta^n}\left(\tilde{\mathcal{X}}(t^n; t^{n+1}, x)\right),$$

with $\tilde{\mathcal{X}}$ an approximate characteristic curve. To satisfy this property, we compute

$$\theta^{n+1} = \operatorname*{argmin}_\theta \int_\Omega \left| u_\theta(x) - u_{\theta^n}\left(\tilde{\mathcal{X}}(t^n; t^{n+1}, x)\right) \right|^2 dx.$$



neural SL scheme: update for one sampled point x_k from t^n to t^{n+1}

step 1: starting from ○, compute the foot ● of the approximate characteristic curve

step 2: fit $u_{\theta^{n+1}}(x_k)$ to match $u_{\theta^n}\left(\tilde{\mathcal{X}}(t^n; t^{n+1}, x_k)\right)$

2.3 - Error bounds

With exact characteristics, the following error estimate holds:

$$\forall n \geq 0, \quad \left\| u_{\theta^n} - u(t^n, \cdot) \right\|_{L^2(\Omega)} \leq \sum_{m=0}^{n} \left[\varepsilon_{\text{int}}^m + \varepsilon_{\text{opt}}^m + \varepsilon_{\text{approx}}^m \right] + \mathcal{O}(\Delta t^p).$$

⇝ accumulation of integration, optimization and approximation errors

2.4 - Algorithm

All optimizations are performed with the Gauss-Newton algorithm.
At each step of the (natural) gradient descent, K points $x_k, \mu_k \in \Omega \times \mathbb{P}$ are sampled.

1. initialization: compute the initial parameters θ^0 by solving

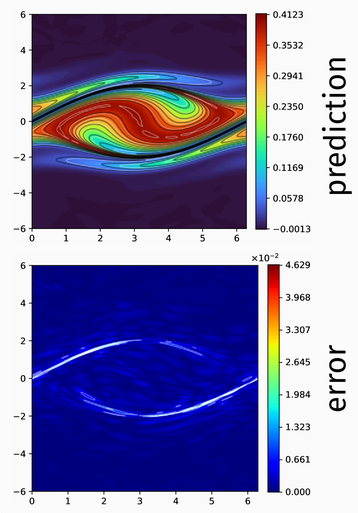
$$\theta^0 = \operatorname*{argmin}_\theta \sum_{k=1}^{K} \left| u_\theta(x_k, \mu_k) - u_0(x_k, \mu_k) \right|^2$$

2. time loop: for each $n \geq 0$,
   a. (approximately) solve the characteristic ODE for all k to find $\tilde{\mathcal{X}}(t^n; t^{n+1}, x_k, \mu_k)$
   b. compute the parameters θ^{n+1} by solving

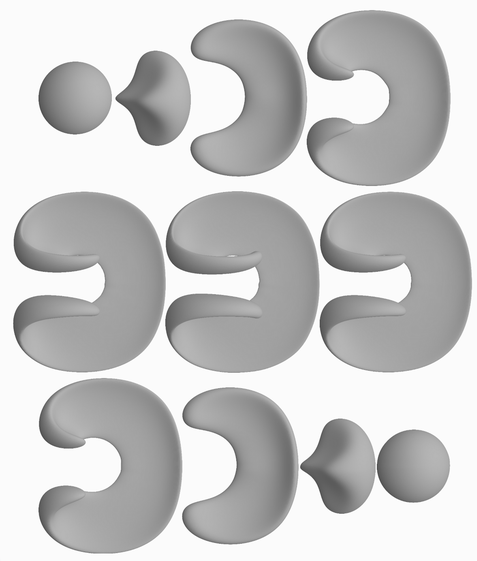
$$\theta^{n+1} = \operatorname*{argmin}_\theta \sum_{k=1}^{K} \left| u_\theta(x_k, \mu_k) - u_{\theta^n}\left(\tilde{\mathcal{X}}(t^n; t^{n+1}, x_k, \mu_k)\right) \right|^2$$

3.1 - 1D1V Vlasov

$$\partial_t u + v\, \partial_x u + \sin(x)\, \partial_v u = 0$$



3.2 - 5D level-set advection



3D transport of a level-set with 2 parameters:

$$\partial_t u + a \cdot \nabla_x u = 0$$

the solution goes back to the initial condition at the final time

3.3 - Extension to advection-diffusion equations

For the advection-diffusion equation $\partial_t u + a \cdot \nabla_x u - \sigma \Delta u = 0$, there are now 2d additional approximate characteristic curves to compute and follow.

In the table, dofs are mesh points for classical SL and NN parameters for neural SL.

d	classical SL			neural SL		
	dofs	CPU time	L^2 error	dofs	CPU time	L^2 error
1	36	0.96	1.57e-3	127	29.84	1.56e-3
2	1444	2.19	1.63e-3	463	33.07	1.54e-3
3	64 000	3.31	1.56e-3	1009	39.45	1.54e-3
4	2.56×10^6	17.09	1.49e-3	1765	50.29	1.55e-3
5	2.43×10^7	50.64	9.11e-3	2731	76.76	1.52e-3
6	4.70×10^7	110.62	3.91e-3	3907	115.22	1.53e-3
7	6.27×10^7	158.15	1.00e-2	5293	172.53	1.54e-3
8	1.00×10^8	271.33	1.66e-2	6889	254.38	1.57e-3

4 - Summary

Summary of the neural SL method:
- fully meshless
- stable without a time step condition
- able to treat high dimension
- applicable to advection-diffusion problems with minimal overhead

Perspectives and follow-up work:
- nonlinear PDEs (Vlasov-Poisson is done, Navier-Stokes is ongoing)
- application to kinetic relaxation
- volume-preserving extension
- rigorous error analysis

References

[1] A. Staniforth and J. Côté, "Semi-Lagrangian Integration Schemes for Atmospheric Models—A Review," *Mon. Weather Rev.*, vol. 119, no. 9, pp. 2206–2223, 1991.

[2] E. Franck, V. Michel-Dansac, L. Navoret, and V. Vigon, "Neural semi-Lagrangian method for high-dimensional advection-diffusion problems," *Comput. Methods Appl. Mech. Engrg.*, vol. 448, no. B, p. 118481, 2026.

E. Franck, V. Michel-Dansac, L. Navoret et V. Vigon. Neural semi-Lagrangian method for high-dimensional advection-diffusion problems. Computer Methods in Applied Mechanics and Engineering, 448(B):118481, 2026

implemented in `scimba`: https://www.scimba.org/