A DISTANCE

A.1 DISTANCE BETWEEN DYNAMICS

We here give the definition of the distance d. Let u and v be two functions of $\mathcal{L}^2(\mathbb{R}^p, \mathbb{R}^p)$. We consider the distance:

$$\mathrm{d}(u, v) = \mathbb{E}_{X\sim p_X} \|u(X) - v(X)\|_2 \tag{18}$$

Naturally, eq. (18) verifies the triangle inequality, the symmetry and the positiveness. Moreover, in this case, for all functions f, $\mathrm{d}(., f)$ is convex. Indeed, for u, v two functions, and $\lambda \in [0, 1]$:

$$\begin{aligned} d(\lambda u + (1-\lambda)v, f) &= \mathbb{E}_{X\sim p_X} \|\lambda u(X) + (1-\lambda)v(X) - f(X)\|_2 \\ &= \mathbb{E}_{X\sim p_X} \|\lambda u(X) - \lambda f(X) - (1-\lambda)f(X) + (1-\lambda)v(X)\|_2 \\ &\leq \lambda \mathbb{E}_{X\sim p_X} \|u(X) - f(X)\|_2 + (1-\lambda)\mathbb{E}_{X\sim p_X} \|v(X) - f(X)\|_2 \end{aligned}$$

Hence the convexity of $\mathrm{d}(., f)$. This consideration suffices to ensure the convexity of $\mathcal{S}_k$ and $\mathcal{S}_u$ defined in section 3.

A.2 DISTANCE BETWEEN FLOWS

Consider the ODE with $X(t), X_0 \in \mathbb{R}^p$:

$$\frac{dX(t)}{dt} = f(X(t)), \quad X(t=0) = X_0 \tag{19}$$

Equation (19) admits a unique solution as soon as f is Lipschitz. We note $X^\star$ this solution. Then, we can defined the flow ϕ_f of such ODE as :

$$\begin{aligned} [0,T]\times\mathbb{R}^p &\to \mathbb{R}^p \\ t, \quad X_0 &\to \phi_f(t, X_0) = X^\star(t) \end{aligned} \tag{20}$$

With the definition of eq. (20), we can define the distance between two flows of ODE as:

$$d_\phi(\phi_u, \phi_f) = \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_f(t, X_0)\| \, dt \tag{21}$$

d_ϕ is positive and symmetric. Let ϕ_u, ϕ_v be two flows, we have the triangle inequality:

$$\begin{aligned} d_\phi(\phi_u, \phi_f) &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_f(t, X_0)\| \, dt \\ &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_v(t, X_0) + \phi_v(t, X_0) + \phi_f(t, X_0)\| \, dt \\ &\leq \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\phi_u(t, X_0) - \phi_v(t, X_0)\| + \|\phi_v(t, X_0) + \phi_f(t, X_0)\| \, dt \\ &\leq d_\phi(\phi_v, \phi_v) + d_\phi(\phi_v, \phi_f) \end{aligned}$$

Let ϕ_f be fixed, we also have the convexity of $d_\phi(., \phi_f)$ with respect to the first argument. Indeed for $\lambda \in [0, 1]$:

$$\begin{aligned} \mathrm{d}_\phi(\lambda\phi_u + (1-\lambda)\phi_v, f) &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\lambda\phi_u(t, X_0) + (1-\lambda)\phi_v - \phi_f(t, X_0)\| \, dt \\ &= \mathbb{E}_{X_0\sim p_{X_0}} \int_{t_0}^{\tau} \|\lambda\phi_u(t, X_0) + (1-\lambda)\phi_v - \lambda\phi_f(t, X_0) - (1-\lambda)\phi_f(t, X_0)\| \, dt \\ &\leq \lambda \, \mathrm{d}_\phi(\phi_u, \phi_v) + (1-\lambda)\mathrm{d}_\phi(\phi_v, \phi_f) \end{aligned}$$

However, in this case the convexity is not ensured with respect to u and vThis is the reason why for theoretical investigations, we consider the distance d instead of d_ϕ.

Nonetheless, $\mathrm{d}_\phi(\phi_u, \phi_f) = 0 \implies \phi_u = \phi_f \implies u = f$.

B REMARK ON ADDITIVE DECOMPOSITION

First, note that in the case of a metric space the decomposition as defined in eq. (2) always exists.

We now detail an intuition for the well-posedness of such decomposition.

Let $\mathcal{H}_k$ be a closed convex subset of functions of an Hilbert space $(E, <, >)$, and f the function we want to approximate with partial knowledge (represented by the space of hypothesis $\mathcal{H}_k$). Then, thanks to Hilbert projection lemma, we have the uniqueness of the minimizer of $\min_{g \in \mathcal{H}_k} \|f - g\|$, i.e it exists one unique $h_k \in \mathcal{H}_k$ such that: $\forall g \in \mathcal{H}_k, \|f - h_k\| \leq \|f - g\|$.

Finally, the additive decomposition hypothesis presents a remarkable advantage in the case of a vector space. Indeed, if $\mathcal{H}_k$ is a (closed) vector space, let $\mathcal{H}_k^{\perp}$ be its supplementary in E, then we have the uniqueness in the decomposition: $f = f_{\mathcal{H}_k} + f_{\mathcal{H}_k^{\perp}}$, where $f_{\mathcal{H}_k^{\perp}} \in \mathcal{H}_k^{\perp}$ and $f_{\mathcal{H}_k} \in \mathcal{H}_k$.

The existence and uniqueness flowing directly from the additive decomposition hypothesis, this can explain why such assumption is common when bridging ML and MB hypothesis.

C UPPER BOUNDS

C.1 DERIVATION OF EQUATION (4)

The first upper bound is a simple use of the triangle inequality:

$$\begin{aligned} \mathrm{d}(h, f) &= \mathrm{d}(h, f) + \mathrm{d}(h_k, f) - \mathrm{d}(h_k, f) \\ &\leq \mathrm{d}(h_k, f) + |\mathrm{d}(h, f) - \mathrm{d}(h_k, f)| \\ &\leq \mathrm{d}(h_k, f) + \mathrm{d}(h, h_k) \end{aligned}$$

C.2 DERIVATION OF EQUATION (5)

To derive the second upper bound, we assume that f_k^{pr} comes from an overall dynamics f^{pr} obeying the additive decomposition hypothesis of eq. (2) so that f^{pr} and f_k^{pr} verifies: $f^{pr} = f_k^{pr} + f_u^{pr}$. First, with computations similar to eq. (4), we have:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, f^{pr}) + \mathrm{d}(f^{pr}, f) \tag{22}$$

Then:

$$\begin{aligned} \mathrm{d}(h, f^{pr}) &= \mathrm{d}(h, f^{pr}) + \mathrm{d}(h_k, f_k^{pr}) - \mathrm{d}(h_k, f_k^{pr}) \\ &\leq \mathrm{d}(h_k, f_k^{pr}) + |\mathrm{d}(h, f^{pr}) - \mathrm{d}(h_k, f_k^{pr})| \\ &\leq \mathrm{d}(h_k, f_k^{pr}) + |\mathrm{d}(h, f^{pr}) - \mathrm{d}(h, f_k^{pr}) - \mathrm{d}(h, f_k^{pr}) + \mathrm{d}(h_k, f_k^{pr})| \\ &\leq \mathrm{d}(h_k, f_k^{pr}) + |\mathrm{d}(h, f^{pr}) - \mathrm{d}(h, f_k^{pr})| + |\mathrm{d}(h_k, f_k^{pr}) - \mathrm{d}(h, h_k)| \\ &\leq \mathrm{d}(h_k, f_k^{pr}) + \mathrm{d}(f^{pr}, f_k^{pr}) + \mathrm{d}(h, h_k) \end{aligned} \tag{23}$$

Combining Equations (22) and (23), we retrieve eq. (5):

$$\mathrm{d}(h, f) \leq \mathrm{d}(h_k, f_k^{pr}) + \mathrm{d}(h, h_k) + \mathrm{d}(f^{pr}, f_k^{pr}) + \mathrm{d}(f^{pr}, f) \tag{24}$$

and we have: $\Gamma = \mathrm{d}(f^{pr}, f_k^{pr}) + \mathrm{d}(f^{pr}, f)$. Γ is a constant of the problem that cannot be optimized.

C.3 UPPER-BOUND USING AUXILIARY DYNAMICS f^{pr}

Let f^{pr} be the dynamics of model data, we can link up the error made by h on true data (following dynamics f) and the error made by h on model data (with dynamics f^{pr}) via:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, f^{pr}) + \mathrm{d}(f^{pr}, f) \tag{25}$$

Thus a pre-training on auxiliary data of dynamics f^{pr} amounts to control the term $\mathrm{d}(h, f^{pr})$ in the upper-bound of eq. (25).

C.4 SELF-SUPERVISION

Let $h = h_k + h_u$ be the function to learn and G_ψ the recognition network providing an estimate $\hat{\theta}_k^i$ of the parameters from an initial sequence $(X_{t_0}^i, \ldots, X_{t_0+k\Delta t}^i)$. This learning setting corresponds to how velocity fields are learned from consecutive measurements of the tracer fields T in section 4.2.

To compute $\mathrm{d}(h_k, f_k^{pr})$ in the case where $f^{pr} = h^\star$, where $h^\star = h_k^\star + h_u^\star$ is a learned model, we rely on the computed θ_k associated to $h_k^\star$ (thanks for example to the algorithm of section 3.2 associated to eq. (4)) to generate a synthetic dataset with achievable supervision in the space of the parameters θ_k.

From a real initial sequence $(X_{t_0}^i, \ldots, X_{t_0+k\Delta t}^i)$, we can estimate the unknown parameter θ_k^i associated to sequence i with the recognition network $G_\psi^\star$ learned with $h^\star$, i.e $\theta_k^i = G_\psi^\star\big(X_{t_0}^i, \ldots, X_{t_0+k\Delta t}^i\big)$. Then, integrating from the initial condition $X_{t_0}^i$, we generate a trajectory of known parameters θ_k^i with dynamics $h^\star$ denoted by: $\tilde{X}^i = \big(\tilde{X}_{t_0}^i, \ldots, \tilde{X}_{t_n}^i\big)$. Sampling the space of initial conditions, we obtain a synthetic dataset: $\big((\tilde{X}^1, \theta_k^1), \ldots, (\tilde{X}^m, \theta_k^m)\big)$ enabling us to perform self-supervision for G_ψ. Let $\hat{\theta}_k^i$ be the parameters estimated by G_ψ from the simulated $\big(\tilde{X}_t^i, \ldots, \tilde{X}_{t+k\Delta t}^i\big)$, we make the following approximation:

$$\mathrm{d}(h_k, f_k) \approx \frac{1}{m} \sum_{i=1}^{m} \left\| \hat{\theta}_k^i - \theta_k^i \right\|_2 \tag{26}$$

D PROOFS

D.1 NOTE ON THE CONVEXITY OF $\mathcal{S}_k$ AND $\mathcal{S}_u$

Convexity of $\mathcal{S}_k$

Proof. Let $u, v \in \mathcal{S}_k$:

$$\begin{aligned} \mathrm{d}(tu + (1-t)v, f) = \|tu + (1-t)v - f\| &= \|tu - tf + (1-t)v - (1-t)f\| \\ &\leq t\mu_1 + (1-t)\mu_1 = \mu_1 \end{aligned}$$

Hence the convexity of $\mathcal{S}_k$. □

Convexity of $\mathcal{S}_u$

Proof. Let $t \in [0, 1]$ and $u, v \in \mathcal{S}_u$.

$$\begin{aligned} \mathrm{d}(h_k, h_k + tu + (1-t)v) &= \mathrm{d}(0, tu + (1-t)v) \\ &\leq t\mathrm{d}(u, 0) + (1-t)\mathrm{d}(v, 0) \\ &\leq \mu_2 \end{aligned}$$

Hence the convexity of $\mathcal{S}_u$. □

D.2 ODE IDENTIFICATION

Consider the following set: $\mathcal{S}_A = \{X(t) \in \mathcal{C}^1([0, T], \mathbb{R}^p) \text{ such that: } \exists A \in \mathcal{M}_{p,p}(\mathbb{R}), X' = AX\}$, where $T > 0$.

$\mathcal{S}_A$ is not a convex set. Consider u and v in $\mathcal{S}_A$, and consider A^u and A^v so that $u'(t) = A^u u(t)$ and $v'(t) = A^v v(t)$. For $\lambda \in]0, 1[$: we have:

$$\begin{aligned} [\lambda u + (1-\lambda)v]' &= \lambda u' + (1-\lambda)v' \\ &= \lambda A^u u + (1-\lambda)A^v v \end{aligned}$$

In general the last term is not equal to $A^{\lambda u + (1-\lambda)v}(\lambda u + (1-\lambda)v)$, for some matrix $A^{\lambda u + (1-\lambda)v}$. Thus $\mathcal{S}_A$ is not a convex set. However, discretizing the trajectories and employing a simple integration scheme leads to considering the following cost function:

$$\mathcal{L}(A) = \sum_t \| \big(X^s(t+1) - (A\Delta t + Id)X^A(t)\big) \|_2^2, \tag{27}$$

As a least square regression problem, $\mathcal{L}(A)$ is convex with respect to A. A least square regression setting can also be recovered using more complex integration schemes, or several time steps integration.

D.3 Proof for Well-posedness of Equation (7)

We set ourselves in the Hilbert space of squared integrable functions with the canonical scalar product $\left(\mathcal{L}^2(\mathbb{R}^p, \mathbb{R}^p), <, > \right)$. For further consideration on such functional space we refer to (Droniou, 2001).

We assume that $\mathcal{H}_k$ hence $\mathcal{S}_k$ is convex and a relatively compact family of functions.

Convexity of $\mathcal{S}_k + \mathcal{S}_u$ Let $\mathcal{S} = \mathcal{S}_k + \mathcal{S}_u = \{f \,|\exists f_k \in \mathcal{S}_k, f_u \in \mathcal{S}_u, f = f_k + f_u\}$.

Let $f, g \in \mathcal{S}$ and $\lambda \in]0, 1[$:

$$\lambda f + (1-\lambda)g = \lambda f_k + (1-\lambda)g_k + \lambda f_u + (1-\lambda)g_u \in \mathcal{S}_k + \mathcal{S}_u$$

Hence the convexity of $\mathcal{S}$.

Closeness of $\mathcal{S}_u$ We show that $\mathcal{S}_u$ is a closed set. Indeed, $\mathcal{S}_u = g^{-1}([0, \mu_u])$, where $g(u) = \|u\|$, Because g is 1-Lipschitz (using the triangle inequality), g is continuous. Therefore $\mathcal{S}_u$ is closed set as the inverse image of a closed set by continuous function.

Sequential Limit We now show that $\mathcal{S}$ is a closed set thanks to the sequential characterisation: let f^n a converging sequence of elements of $\mathcal{S}$ and denote f its limit. We prove that f^n converges in $\mathcal{S}$.

Because $\forall n, f_n \in \mathcal{S}$, we have: $f^n = f_k^n + f_u^n$, where $f_u^n \in \mathcal{S}_u$ and $f_k^n \in \mathcal{S}_k$.

Thanks to the relative compactness of $\mathcal{S}_k$, we can extract a converging sub-sequence, of indexes n_j, from f_k^n so that $f_k^{n_j} \to f_k \in \mathcal{S}_k$.

Because $f^n \to f$, the sub-sequence f^{n_j} converges: $f^{n_j} \to f$.

By definition, $f_u^{n_j}$ is a sequence of $\mathcal{S}_u$ and we also have that: $f_u^{n_j} = f^{n_j} - f_k^{n_j}$. Because the right member of the equation converges (as a sum of converging functions), the left member of the equation converges i.e. $f_u^{n_j}$ converges.

Since $\mathcal{S}_u$ is a closed set $f_u^{n_j}$ converges in $\mathcal{S}_u$. We write f_u its limit. Therefore, $f_u^{n_j} = f^{n_j} - f_k^{n_j} \to f - f_k = f_u \in \mathcal{S}_u$. Hence, $f = f_u + f_k$ with $f_u \in \mathcal{S}_u$ and $f_k \in \mathcal{S}_k$.

Therefore $\mathcal{S}$ is a closed set.

Finally, we can apply Hilbert projection lemma on the closed convex set $\mathcal{S}$ and retrieve the uniqueness of the minimizer of eq. (7).

Remark The relative compactness of a family of functions is a common assumption in functional analysis. For example, in the study of differential equation Cauchy-Peano theorem provides the existence to the solution of an ODE under the assumption of relative compactness.

Also, Ascoli theorem provides the relative compactness of a family of function $\mathcal{F}$ under the hypothesis of the equi-continuity of $\mathcal{F}$ and the relative compactness of the image space $A(x) = \{f(x) | f \in \mathcal{F}\}$.

D.4 Proof of Proposition 2

We now set ourselves in the Hilbert space $\left(\mathcal{L}^2([0, T], \mathbb{R}^p), <, > \right)$ of squared integrable functions, where $<, >$ is the canonical scalar product of $\mathcal{L}^2([0, T], \mathbb{R}^p)$.

Proof. Let A be a given invertible matrix. We consider the following space $\mathcal{S}_D = \{X \in \mathcal{C}^1([0, T], \mathbb{R}^p)$ such that: $\exists D \in \mathbb{R}^p, X' = AX + D$ and $X(t = 0) = X_0\}$, where $T > 0$. We show that $\mathcal{S}_D$ is a closed convex set.

Convexity Indeed, let $\lambda \in]0,1[$ and $u, v \in \mathcal{S}_D$. $\lambda u + (1-\lambda)v$ is differentiable and:

$$[\lambda u + (1-\lambda)v)]' = \lambda u' + (1-\lambda)v' = A(\lambda u + (1-\lambda)v) + D,$$

Where $D = \lambda D_u + (1-\lambda)D_v$. Hence $\lambda u + (1-\lambda)v \in \mathcal{S}_D$.

Closeness via Affine-Space To prove the closeness of $\mathcal{S}_D$, we prove that it is an affine space of finite dimension.

Let g the application that to any vector $D \in \mathbb{R}^d$ associate the solution X^D.

Let $D_0 \in \mathbb{R}^D$, we show that $g_{D_0} : D \to g(D_0 + D) - g(D_0)$ is a linear application.

Naturally, for $g_{D_0}(0_{\mathbb{R}^p}) = 0_{\mathcal{L}^2}$. Then for $D \neq 0_{\mathbb{R}^p}$ we have:

$$\begin{aligned} g_{D_0}(D) &= e^{At}(X_0 + A^{-1}(D_0 + D)) - A^{-1}(D_0 + D) - e^{At}(X_0 + A^{-1}(D_0) + A^{-1}D_0 \\ &= e^{At}A^{-1}D \end{aligned}$$

Therefore g_{D_0} is a linear function and g is an affine function.

Moreover, g is an injection. Indeed, if two functions are equals, then they have at most one inverse image by g thanks to Cauchy-Lipschitz theorem. Therefore it defines a bijection of $\mathbb{R}^d$ in $g(\mathbb{R}^d)$. Since, $\mathcal{S}_D = g(\mathbb{R}^d)$, $\mathcal{S}_D$ is an affine space of dimension p and g is continuous in particular for the canonical norm induced on $\mathcal{L}^2([0,T], \mathbb{R}^p)$. Therefore $\mathcal{S}_D$ is an affine space of finite dimension and is a closed set.

Finding a Unique Minimizer We conclude by applying Hilbert projection lemma: our problem of minimizing $\int_0^T \left\|X^s(\tau) - X^D(\tau)\right\|$, amounts to an orthogonal projection problem. Because $\mathcal{S}_D$ is a closed convex set, we have existence and uniqueness of such projection. Therefore, it exists a unique function $X_D \in \mathcal{S}_D$ and a unique vector D minimizing its distance to the function X^s. □

D.5 Algorithm in Linear Setting

We detail in Algorithm 2 the alternate projection algorithm in a linear setting. We denote $Y = (X^i_{t_0+\Delta t}, X^i_{t_0+n\Delta t})$ and $X = (X^i_{t_0}, X^i_{t_0+(n-1).\Delta t})$. For readability purposes we set $\Delta t = 1$.

Algorithm 2 Alternate estimation: Linear Setting

Result: $A \in \mathcal{M}_{p,p}(\mathbb{R}), D \in \mathbb{R}^p$
$k = 0$, $D^0 = 0$, $A_0^{-1} = 0$ $A_0^0 = min_A \|\mathcal{Y} - XA\|$
while $\|D^k - D^{k-1}\| > \epsilon$ *and* $\|A^k - A^{k-1}\| > \epsilon$ **do**
  $D^{k+1} = \min_D \|\mathcal{Y} - XA^k - D\|_2^2 + \lambda\|D\|_2^2$
  $A^{k+1} = \min_A \|\mathcal{Y} - XA - D^{k+1}\|_2^2 + \gamma\|Y - XA\|_2^2$
  $k \leftarrow k + 1$
end

D.6 Proof to Proposition 3

Naturally, one could estimate jointly D and A using least square regression. However, the idea is to verify the convergence of such alternate algorithm in a simple case. We conduct the proof for the first dimension of $\mathcal{Y}$ to lighten notations, meaning that we are regressing the first dimension of Y against the X.

A similar reasoning for the other dimension completes the proof.

Proof. We first give the analytical solution for D. Let A^n be fixed.

Estimation of D Consider:

$$\mathcal{L}_D = \|Y - XA^n - D\|_2^2 + \lambda\|D\|_2^2 \tag{28}$$

where $D = (d, \ldots, d) \in \mathbb{R}^Q$. For Q samples, we find d so that $\frac{\partial L}{\partial d} = 0$:

$$\begin{aligned}\frac{\partial L}{\partial d} = 0 \Leftrightarrow -2 * \sum_{i=1}^{Q}(y_i - X_i A^n - d) + 2\lambda d = 0 \\ \Leftrightarrow Qd + \lambda d = \sum_{i=1}^{Q}(y_i - X_i A^n) \\ \Leftrightarrow d(Q+\lambda) = \sum_{i=1}^{Q}(y_i - X_i A^n) \\ \Leftrightarrow d = \frac{\overline{Y - XA}}{1 + \lambda/Q}\end{aligned}$$

where $\overline{Y - XA} = \frac{1}{Q}\sum_{i=1}^{Q}(y_i - X_i A^n)$.

Estimation of A Let D be fixed and consider:

$$\mathcal{L}_A = \|Y - XA - D\|_2^2 + \gamma\|Y - XA\|_2^2 \tag{29}$$

Similarly, we aim to cancel the first derivative of $\mathcal{L}_A$ with respect to all parameters of $A = (a_1, .., a_p)$:

$$\begin{aligned}\frac{\partial \mathcal{L}_A}{\partial a_j} = 0 \Leftrightarrow &- 2 * \sum_{i=1}^{Q} x_{i,j}(y_i - a_0 x_{i,0} + \cdots + a_p x_{i,p} - d) \\ &- 2\gamma * \sum_{i=1}^{Q} x_{i,j}(y_i - a_0 x_{i,0} + \cdots + a_p x_{i,p}) = 0 \\ \Leftrightarrow &- 2X^t(Y - XA - D) - 2\gamma X^t(Y - XA) = 0 \\ \Leftrightarrow &(1+\gamma)X^t XA - X^t(Y - D) - \gamma X^t Y = 0 \\ \Leftrightarrow &(1+\gamma)X^t XA = X^t\big(\gamma Y + (Y - D)\big) \\ \Leftrightarrow &A = \frac{B^{-1}X^t}{1+\gamma}\big((1+\gamma)Y - D\big)\end{aligned} \tag{30}$$

where $B = X^t X$. Equation (30) indicates that as soon a D converges, A^n converges. Thus, we now prove the convergence of (D^n). Then, for $n > 1$ consider:

$$\begin{aligned}\big\|D^{n+1} - D^n\big\| &= \frac{1}{1+\lambda/Q}\Big\|\overline{Y - XA^n} - \overline{Y - XA^{n-1}}\Big\| \\ &= \frac{1}{1+\lambda/Q}\Big\|\overline{X(A^n - A^{n-1})}\Big\| \\ &= \frac{1}{(1+\lambda/Q)(1+\gamma)}\Big\|\overline{XB^{-1}X^t\big([(1+\gamma)Y - D^n] - [(1+\gamma)Y - D^{n-1}]\big)}\Big\| \\ &= \frac{1}{(1+\lambda/Q)(1+\gamma)}\Big\|\overline{XB^{-1}X^t[D^{n-1} - D^n]}\Big\| \\ &\leq \frac{K}{(1+\lambda/Q)(1+\gamma)}\big\|D^{n-1} - D^n\big\|\end{aligned}$$

where $K = \|XB^{-1}X^t\|$.

Therefore, for λ, γ, sufficiently large, $\frac{K}{(1+\lambda/Q)(1+\gamma)} < 1$. $\|D_n - D_{n-1}\|$ converges as a positive decreasing sequence. Finally, the sequence of (D_n) converge and so the sequence of (A_n).

In conclusion, the proposed algorithm converges. □

E DATASETS

In this section, we provide exhaustive simulation details for the damped pendulum, Lotka-Volterra, and both geophysical datasets.

E.1 DAMPED-PENDULUM

For the damped pendulum data, eq. (15) is integrated with $\Delta t = 0.2s$ using a Runge-Kutta 4-5 scheme from $t = 0$ up to $t = 10s$. Both the pulsation ω_0 and the damping coefficient k are fixed across the dataset. We generate 100/50/50 sequences respectively for train, validation and test sampling over the initial conditions so that $(\theta, \dot{\theta}) \sim \mathcal{U}\big([-\pi/2, \pi/2] \times [-0.1, 0.1]\big)$.

Small Oscillations To linearize the pendulum, we consider the small oscillations regime and take the initial conditions so that : $(\theta, \dot{\theta}) \sim \mathcal{U}\big([-\pi/6, \pi/6] \times [-0.1, 0.1]\big)$. In that case eq. (15) writes as:

$$\frac{d}{dt}\begin{pmatrix}\dot{\theta}\\ \theta\end{pmatrix} = \begin{pmatrix}-\lambda & \frac{g}{L}\\ 1 & 0\end{pmatrix}\begin{pmatrix}\dot{\theta}\\ \theta\end{pmatrix} \tag{31}$$

and following notations of section 3.3, we have: $D_A = 0$ and $A = \begin{pmatrix}-\lambda & \frac{g}{L}\\ 1 & 0\end{pmatrix}$

E.2 LOTKA-VOLTERRA

For Lotka-Volterra data, eq. (16) is integrated with $\Delta t = 0.05$ using a Runge-Kutta 4-5 scheme from $t = 0$ up to $t = 20$. All parameters $\alpha, \beta, \gamma, \delta$ are set to 1 across the dataset. We generate 100/50/50 sequences respectively for train, validation and test sampling over the initial prey and predators populations so that $(x, y) \sim \mathcal{U}\big([0, 2]^2\big)$.

Practical Issues and Adaptation Assuming that α and γ have positive values makes the following problem arises: the trajectories defined by h_k for the prey are unbounded, whereas the trajectories defined by eq. (16) are. Minimizing $\mathrm{d}(h_k, f)$ over long term horizon will lead in an underestimation of α to match the bounded behaviour of true data. Therefore, we enforce $\mathrm{d}(h_k, f)$ on the prey component as soon as the number of predator is small. In practice, we set this threshold to 0.15.

E.3 GEOPHYSICAL DATASETS

We present in this section introductory tools for the understanding of the fluid dynamics data presented in section 4.2. We first introduce the physical modeling of ocean dynamics. Then, we outline the Adv+S dataset simulation which draws from ocean modeling. Finally, we introduce the Natl dataset and the proxy data used in the experiments.

Introduction To Ocean Modeling The increase in ocean observations thanks to satellites and floats enabled a great development in Earth modeling over the last decades. The ocean circulation, that is the current velocity fields dynamics, are now realistically modeled in tri-dimensional structured models such as NEMO (Madec, 2008).

Such models rely on in-depth physical knowledge of the studied system and its representation through partial differential equations. Integrated over depth, the equations associated to the transport of the Sea Surface Temperature T by a time-varying horizontal velocity field U can be written as:

$$\frac{\partial T}{\partial t} = -\nabla.(TU) + D^T + F^T \tag{32}$$

$$\frac{\partial U}{\partial t} = -(U.\nabla)U + f \wedge U - g'\nabla h + D^U + F^U \tag{33}$$

where f is the Coriolis parameter, h the depth of the surface layer obtained from sea surface height (SSH) observations, g' the reduced gravity which takes the stratification in density of the ocean into account such that $g' \approx g.10^{-3}$. In a two-dimensional setting, $\nabla(TU)$ refers to the advection of a

scalar quantity T by a velocity field $U = (u, v)$ and writes as : $\nabla(TU) = \frac{\partial T}{\partial x}u + \frac{\partial T}{\partial y}v$. The mixing terms, referred to as $D^{T/U}$ and the forcings $F^{T/U}$, are not known.

In the context of the presented work, the physical state $Z_t = (T_t, U_t)$, f_X and f_Y from eq. (1) can be interpreted as follows: f_X represents the dynamics of the observed T, i.e. $f_X(T) = -\nabla.(TU) + D^T + F^T$ in eq. (32). f_Y represents the dynamics of U in eq. (33), i.e. $f_Y(U, h) = -(U.\nabla)U + f \wedge U - g'\nabla h + D^U + F^U$.

Whereas T is observed by satellites, U is not known. However, the Sea Surface Height (SSH) could be used to compute coarse estimates of U. Indeed, under hypothesis such as stationarity ($\frac{\partial U}{\partial t} = 0$), incompressibility ($(U.\nabla)U = 0$)), forcings can be omited. In this case, eq. (33) can be rewritten into

$$f \wedge U = -g'\nabla h \tag{34}$$

When projected onto x and y axis, eq. (34) becomes

$$-fv = -g'\frac{\partial h}{\partial x}, \qquad fu = -g'\frac{\partial h}{\partial y}, \tag{35}$$

Note that eq. (34) and eq. (35) do not hold at fine scales as the stationarity and incompressibility assumptions only hold at large scale. In this case, the SSH h can be regarded as a stream function.

Both datasets considered in the paper follow the same equations approximating the tracer equation (eq. (17)) inspired by eq. (32):

$$\frac{\partial T}{\partial t} = -\nabla.(TU) + S \tag{36}$$

We study the equations 32 and 33 in an incremental approach. In the following parts, we describe how T, U and S are computed in both datasets Adv+S and Natl.

E.3.1 ADV+S

We first investigate a dataset generated following simplifying assumptions (Adv+S). We don't rely on true U and S, we instead build them so that they correspond to our hypothesis.

Building a Velocity Field U Under stationarity and incompressibility hypothesis, U can be approximated from a stream function $\mathcal{H}$. Note that, in this dataset, $\mathcal{H}$ is not equal to the SSH h, it is simulated following (Boffetta et al., 2001):

$$\mathcal{H}(x, y, t) = -\tanh[\frac{y - B(t) \times \cos kx}{\sqrt{1 + k^2 B(t)^2 \times sin^2 kx}}] + cy, \tag{37}$$

As introduced precedently (see eq. (34)), eq. (33) can be simplified and we compute $U = (u, v)$ so that it follows:

$$u = -\frac{\partial \mathcal{H}}{\partial y}, \quad v = \frac{\partial \mathcal{H}}{\partial x} \tag{38}$$

Note that B varies periodically with time according to $B = B_0 + \epsilon \cos(\omega t + \phi)$. We compute 10 different velocity fields sampling random parameters $B_0, k, c, \omega, \epsilon, \phi$.

Building a Source Term S In eq. (32), the diffusion term D^T is omitted. We generate the source term S so that it represents the forcing term F^T in eq. (36). To illustrate heat exchanges, we draw from Frankignoul (1985). This source term is a non linear transformation of $U = (u, v)$ multiplied by the difference between the ocean temperature and a reference temperature:

$$S(U, T) = w_e \times (T - T_e) \quad \text{where} \quad w_e = \begin{cases} 0 & \text{if } \frac{\partial \mathcal{H}}{\partial t} < 10^{-4} \\ 1 & \text{otherwise.} \end{cases}$$

where T_e is the sequence mean image (computed without source).

Dataset Generation Using computed U and S, we integrate eq. (36) with $\Delta t = 8640s$ over 30 days, using a Semi-Lagrangian scheme (see explanations below). We generate 800/100/200 sequences respectively for train, validation and test sampling over the initial conditions, which are images of size 64×64 sampled from Natl dataset. Finally, for integration, we impose East-West periodic conditions, implying that what comes out the left part re-enters at the right, and reciprocally. We also impose velocity to be null on both top and bottom parts of the image.

Semi-Lagrangian Integration Unlike Eulerian scheme, relying on time discretization of the derivative, the semi Lagrangian scheme relies on the constancy of the solution of a PDE along *a characteristic curve*. Consider a solution to the advection equation, i.e. eq. (36) with $S = 0$. The method of characteristics consists in exhibiting curves $(x(s), t(s))$ along which the derivative of the solution T is simple, i.e $\frac{\partial T}{\partial s}(x(s), t(s)) = 0$. For a 1D constant advection scheme, computations lead to:

$$\frac{dt}{ds} = 1 \implies s = t$$
$$\frac{dx}{ds} = U \implies x = x_0 + Ut$$

giving therefore, $T(x, t) = T_0(x - Ut)$, linking the value of the solution at all time to its initial condition. Therefore from a single observation at t_0, it suffices to estimate the original departure points $x_0 - Ut$ to infer the prediction at t.

However, when U is not constant in time, the method remains doable, not along characteristic *lines* defined by : $(x_0 + Ut)$, but along characteristic *curves* which are given by:

$$\frac{dt}{ds} = 1 \implies s = t$$
$$\frac{dx}{ds} = U(x, t) \tag{39}$$

A great deal in the semi-Lagrangian literature involves solving correctly eq. (39). We use the conventional mid-point integration rule and the semi-Lagrangian is implemented using Pytorch function `gridsample`, following in (Jaderberg et al., 2015). Further developments can be found for example in (Diamantakis, 2014).

E.3.2 NATL

This second dataset depicts the actual ocean circulation, i.e. we consider both eq. (32) and eq. (33). In this case, no assumptions are made on U and S represents both diffusion terms D^T and forcing terms F^T. We access daily data over a year of ocean surface temperature of the North Atlantic observations model resulting from (Ajayi et al., 2019) [1]. The dataset covers a 2300km $\times$ 2560km zone at 1.5km resolution, in the North Atlantic Ocean.

In this real-life dataset, sea surface height (SSH) partial derivative with respect to x and y serves as proxies to the (unobserved) velocity fields U. Indeed, recall that simplifying hypotheses led us to eq. (35).

We divide the Natl zone into 270 patches of size 64×64. For each region, we extract sea surface temperatures, velocity fields, source terms and height variables. We sample 200/20/50 sequences of 1 year, for respectively train, validation and test. In this case, $\Delta t = 86400s$ (1 day).

F TRAINING DETAILS

All experiments were conducted on NVIDIA TITAN X GPU using Pytorch (Paszke et al., 2019).

Hyper-Parameters Interpretation From eq. (4), two independent terms appear justifying an alternate projections approach.

First, we highlight that strictly minimizing $\mathrm{d}(h_k, f)$ biases our estimation of h_k. However, it may yield a good estimation of h_k provided that f_k contributes significantly to the prediction of f. Hence, we interpret this loss as an *initialization* loss. Thus, in most applications, we progressively decrease its magnitude along training as detailed in appendices F.1 to F.3.

On the other hand, $\mathrm{d}(h_u, 0)$ aims at constraining the free form function h_u to make its action as small as possible. We interpret this loss as a *stability* penalty.

[1] Details available at : https://meom-group.github.io/swot-natl60/access-data.html

Finally, aiming to recover exact trajectories of observations, we proceed as suggested in (Yin et al., 2021) progressively increasing the hyper-parameters associated to $\mathrm{d}(h, f)$.

The practical implementation is summarized in the following algorithm:

Algorithm 3 Alternate estimation: Practical Setting

Initialization: $\theta_u^0 = 0,\ \theta_k^0 = \min_{h_k \in \mathcal{H}_k} \mathrm{d}(h_k, f)$, λ_h, λ_{h_k}, λ_{h_u}
for $epoch = 1 : N_{epochs}$ **do**
  for $batch = 1 : B_k$ **do**
    $$\theta_k^{n+1} = \theta_k^n - \tau_1 \nabla_{\theta_k} [\lambda_h \mathrm{d}(h, f) + \lambda_{h_k} \ell(h_k)]$$
  end
  for $batch = B_k : B_u$ **do**
    $$\theta_u^{n+1} = \theta_u^n - \tau_1 \nabla_{\theta_u} [\lambda_h \mathrm{d}(h, f) + \lambda_{h_u} \mathrm{d}(h_u, 0)]$$
  end
  $\lambda_h = \tau_2 \lambda_h$; $\lambda_{h_k} = \frac{1}{\tau_2} \lambda_{h_k}$; $\lambda_{h_u} = \frac{1}{\tau_2} \lambda_{h_u}$
end

F.1 Damped Pendulum

Architecture Details The physical parameters to be learned is a scalar of dimension 1, and h_u is a 1-hidden layer MLP with 200-hidden neurons with leaky-relu activation.

Optimization For this dataset we use RMSProp optimizer with learning rate 0.0004 for 100 epochs with batch size 128. We supervise the trajectories up to $t = \Delta t \times 50$, i.e we enforce d_ϕ over $(t_0 + \Delta t, .., t_0 + 50\Delta t)$. Overall the number of optimization subsequences for training is 17000. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 10-batches on h_k then 10-batches on h_u.

Hyperparameters We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.001$. We initialize $\lambda_h = 0.1$ and increase it geometrically up to $\lambda_h = 100$. λ_{h_u} is fixed through training at 0.1.

The hyper-parameters were chosen by randomly exploring the hyper-parameters space by sampling them so that $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$. We select the ones with the lowest prediction errors, i.e with lowest $\mathrm{d}_\phi(h, f)$.

For the ablation study of Table 1, we set to 0 the hyper-parameters associated to the non-considered loss.

The training time for this dataset is 1 hour.

F.2 Lotka-Volterra

Architecture Details The physical parameters to be learned is a vector of dimension 2 accounting for (α, β) in eq. (16), and h_u is a 1-hidden layer MLP with 200-hidden neurons with leaky-relu activation.

Optimization We use Adam optimizer with learning rate 0.0005 for 200 epochs with batch size 128. Overall the number of sequences for training is 15000. We supervise the trajectories up to $t = \Delta t \times 25$, i.e we enforce d_ϕ over $(t_0 + \Delta t, .., t_0 + 25\Delta t)$. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 10-batches on h_k then 10-batches on h_u.

Hyperparameters We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.001$. We initialize $\lambda_h = 0.001$ and increase it geometrically up to $\lambda_h = 1$. λ_{h_u} is fixed through training at 0.001.

The hyper-parameters were chosen by randomly exploring the hyper-parameters space by sampling them so that $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$. We select the ones with the lowest prediction errors (i.e lowest $\mathrm{d}(h, f)$).

For the ablation study of Table 1, we set to 0 the hyper-parameters associated to the non-considered loss.

The training time for this dataset is 2 hours.

F.3 ADV+S

Architectures Details The physical parameters to be estimated are the velocity fields U, of dimension $(2, 64, 64)$. As U varies over time, we follow data assimilation principles to map a sequence of 4 consecutive measurements of the tracer field T to the associated velocity field (Gaultier et al., 2013). To do so, we parameterize a recognition network G_ψ by U-net with at most 512 latent channels also following the implementation of (Isola et al., 2017), taking as input a sequence of 4 time steps of T: $(T_{t_0}, .., T_{t_0+3\Delta t})$. The residual dynamics h_u is learned by a convolutional ResNet, with 1 residual block taking as entry the same sequence of T. We implement h_k via a semi-lagrangian scheme, taking as input T_t and the estimated U_t to predict T_{t+1}.

Optimization We use Adam optimizer with learning rate 0.0001 for 30 epochs with batch size 32. We supervise the trajectories up to $t = \Delta t \times 6$, i.e we enforce d_ϕ on $(T_{t_0+\Delta t}, ..., T_{t_0+6\Delta t})$. Overall the number of sequences for training is 36800. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 4-batches on h_k then 6-batches on h_u.

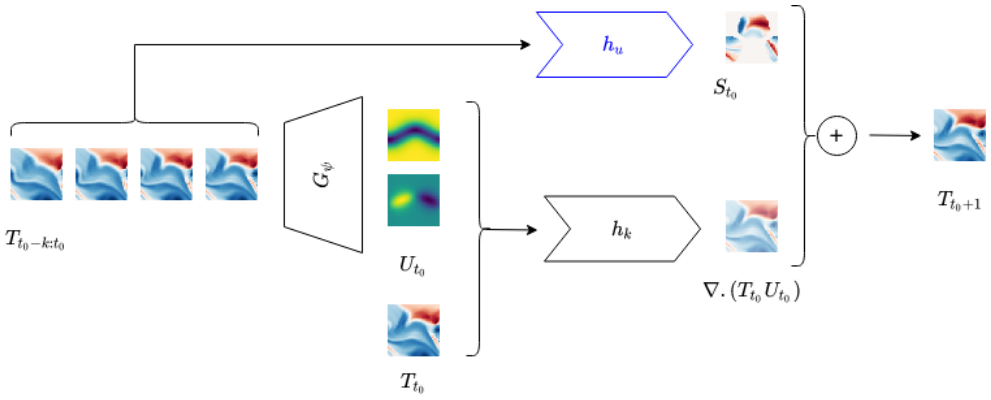

Figure 3: *Best viewed in color*. Schematic view of our model in the context of section 5.2, applied on the Adv+S dataset.

Hyperparameters, setting of eq. (4) We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.00001$. We initialize $\lambda_h = 0.01$ and increase it geometrically every epoch up to $\lambda_{h,f} = 1000$. λ_{h_u} is fixed through training at 0.1. We select the hyperparameters with the lowest prediction errors (i.e lowest $\mathrm{d}(h, f)$). For the ablation study of Table 1, we set to 0 the hyper-parameters associated to the non-considered loss.

The training time for this dataset is 8 hours.

F.4 NATL

Architecture Details The architectures in this setting are identical to the ones described in appendix F.3.

Optimization We use Adam optimizer with learning rate 0.00001 for 50 epochs with batch size 32. Overall the number of sequences for training is 67000. We enforce d_ϕ over 6 time-steps, i.e we supervise the predictions on timesteps: $(t_0 + \Delta t, .., t_0 + 6\Delta t)$. We use dropout in both G_ψ and h_u.

Hyperparameters, setting of eq. (4) For this setting, λ_h geometrically increases from 0.01 up to 100. We initialize $\lambda_{h_k} = 0.1$ and decrease it geometrically down to $\lambda_{h_k} = 0.00001$. λ_{h_u} is fixed through training at 0.1. We alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 10-batches on both h_k and h_u.

The selected model is the one with lowest prediction errors on validation set (i.e lowest $\mathrm{d}(h, f)$), sampling uniformly the hyperparameters: $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$.

Hyperparameters, setting of eq. (5) Because the dynamics of Natl is highly non linear and chaotic, we follow (Jia et al., 2019) and first warm-up the parameters recognition network G_ψ on the velocity fields proxies for 10 epochs. For this setting, λ_h geometrically increase from 0.01 up to 1. λ_{h_k} is set equal to λ_h. λ_{h_u} is fixed through training at 0.01.

After warm-up, we alternate projection on $\mathcal{S}_k$ and $\mathcal{S}_u$ by descending the gradient 100-batches on h_k and 300 on h_u. In this setting of eq. (5), the selected model is the one with lowest $\mathrm{d}(h, f) + \mathrm{d}(h_k, f_k^{pr})$ error, sampling uniformly the hyperparameters: $\lambda \sim \mathcal{U}(1, 0.1, \ldots, 10^{-4})$.

The training time for this dataset is 12 hours.

Baselines We train NODE (Chen et al., 2018) and Aphynity (Yin et al., 2021) on both the Adv+S and Natl dataset. For the training of Aphinity, we set the learning rate at 0.0001 and train on 30 epochs. We initialize $\lambda_h = 0.01$ and increase it geometrically every epoch up to $\lambda_h = 100$. λ_{h_u} is fixed through training at 0.1. For the training of NODE, we set the learning rate at 0.00004 and train on 50 epochs. To perform prediction, we first encode the 4-consecutive measurements of T (as a $3 \times 64 \times 64$ state) then learn to integrate this state in time thanks to a network h. h is a 3-layer convolutional networks, with 64 hidden channels. It is integrated using RK4 scheme available from `https://github.com/rtqichen/torchdiffeq`.

G Additional Results and Samples

G.1 Results for Pendulum and Lotka-Volterra Datasets

We provide respectively in figs. 4 and 5 phase diagrams for the damped pendulum and Lotka-Volterra experiments. Both graphs in the phase space indicate that the trajectories and their nature are well handled by the learned decomposition, providing a periodic phase space for Lotka-Volterra (fig. 5), and a converging spiral for the damped pendulum (fig. 4).

G.2 Results for Adv+S and Natl

In this section, we provide additionial results on both Adv+S and Natl datasets. A thorough ablation study (table 4) gives results with constant hyperparameters λ_h and λ_{h_k} (row Vanilla Optim), which validates our hyper-parameters interpretation. Indeed, the results are better when respectively increasing and decreasing λ_h and λ_{h_k}. Besides, the row Ours eq. (5) refers to a training performed as introduced in appendix C.4 with $f^{pr} = h^\star$ trained on eq. (4). Figure 7 shows predictions up to 4 days on the Adv+S data. Finally, figs. 9 and 11 provide results on Natl dataset associated to training relying on both eq. (4) and eq. (5) and with NODE (Chen et al., 2018).

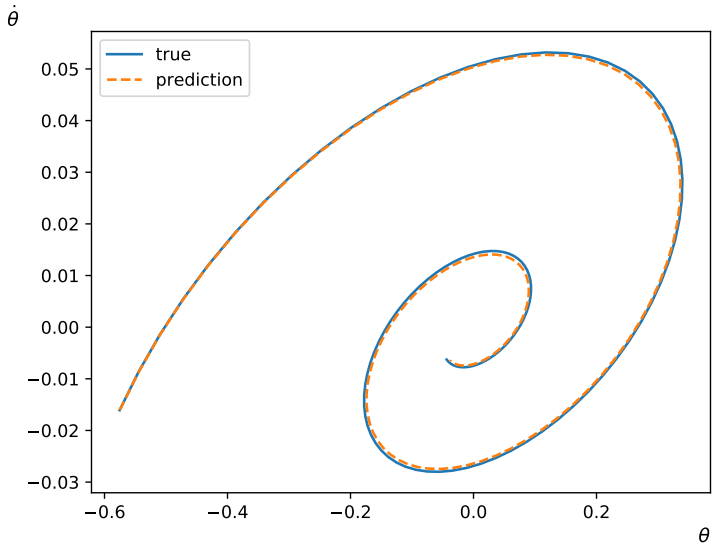

Figure 4: Damped Pendulum Phase Diagram. The true phase diagram (blue) and learned (orange dashed) are close, indicating consistency in the prediction

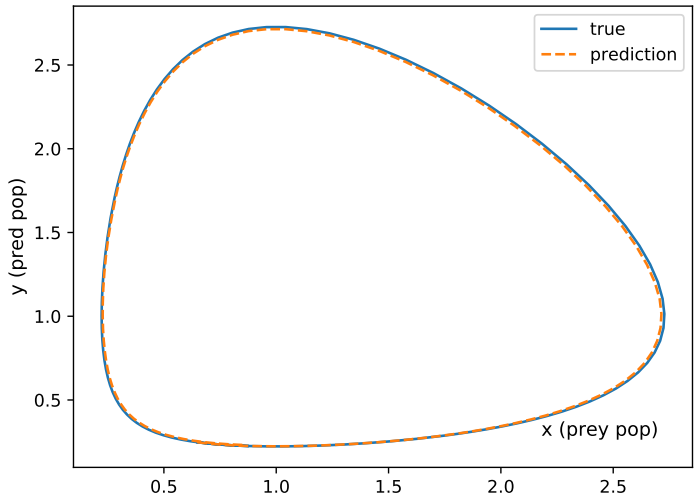

Figure 5: Lotka-Volterra Phase Diagram. The true phase diagram (blue) and learned (orange dashed) are close, indicating consistency in the prediction

Table 4: Ablation Study on Adv+S. We report the MSE ($\times$ 100) on the predicted observations T, the estimated velocity fields U and the residual source term S over 6 and 20 time steps from an initial datum t_0. Unlike alternate training, i.e. Algorithm 1, "Joint" rows refer to the simultaneous optimization of h_k and h_u.

Training	Models	$t_0 + 6$			$t_0 + 20$		
		T	U	S	T	U	S
	Ours (U known)	0.52	n/a	0.19	2.0	n/a	0.32
Alternate	Ours eq. (4)	**0.74**	**1.99**	0.17	8.49	**2.26**	0.31
	only $\mathrm{d}(h, f)$	1.02	4.08	0.19	10.59	4.19	0.32
	$\mathrm{d}(h, f) + \mathrm{d}(h_k, f)$	1.02	3.66	0.19	11.42	3.84	0.34
	$\mathrm{d}(h, f) + \mathrm{d}(h, h_k)$	0.77	2.38	0.19	9.5	2.45	0.34
	Ours eq. (5)	0.75	2.77	**0.17**	**8.36**	2.84	**0.29**
	Vanilla optim.	1.51	3.77	0.3	13.33	4.1	5.15
Joint	Ours eq. (4)	1.44	3.3	0.3	12.82	3.5	0.5
	only $\mathrm{d}(h, f)$	1.38	6.96	0.39	11.9	7.09	0.54

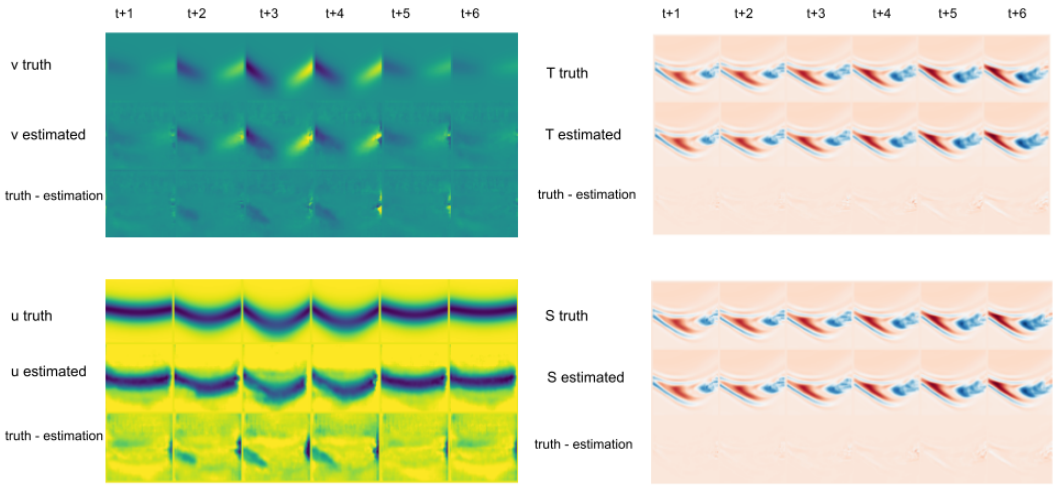

Figure 6: *Best viewed in color.* Estimations, targets and differences between estimations and targets on T, $U = (u, v)$ and S for Adv+S. Each column refers to a time step, ranging from 1 to 6 half-days. On the left, true and estimated $U = (u, v)$ over 6 time steps, and differences between targets and estimations. On the right, prediction of T and S over 6 time steps, and differences between targets and estimations.

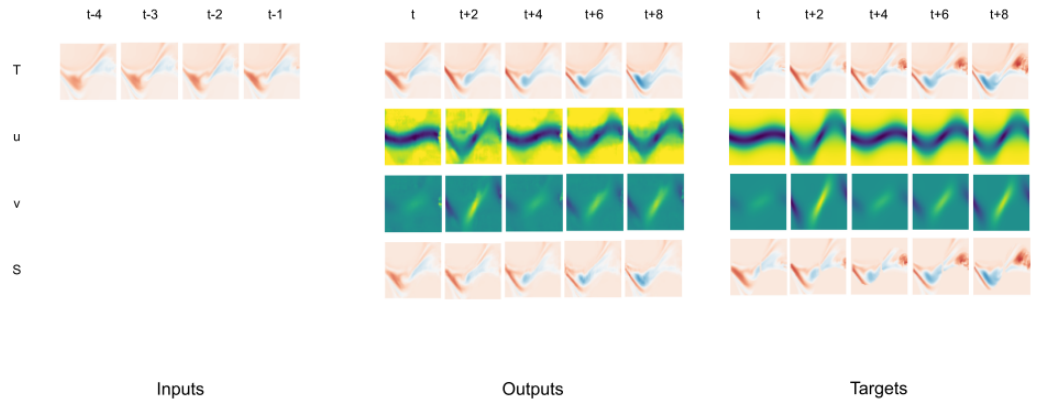

Figure 7: *Best viewed in color*. Estimations and targets on T, $U = (u, v)$ and S for Adv+S. Each column refers to a time step, ranging from 1 to 8 half-days. On the left, sequence of T inputs (4 time steps). In the middle, prediction of T, $U = (u, v)$ and S over 8 time steps. On the right, true T, U and S over 8 time steps.

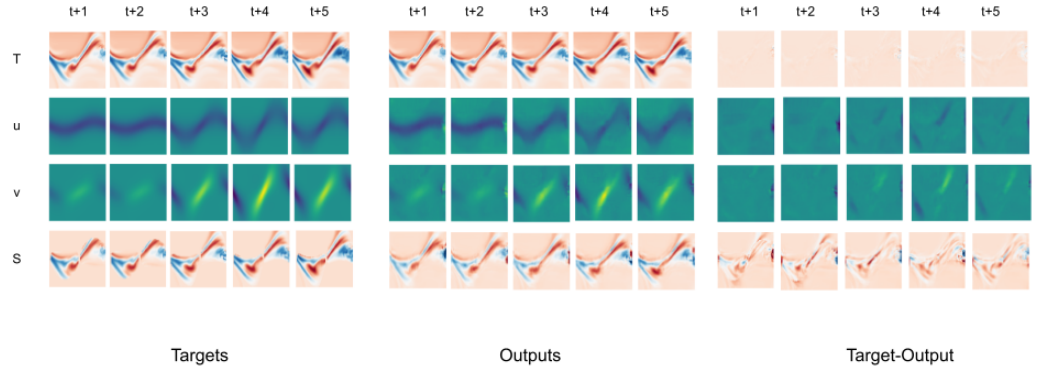

Figure 8: *Best viewed in color*. Estimations, targets and differences between estimations and targets on T, $U = (u, v)$ and S for Adv+S. Each column refers to a time step, ranging from 1 to 5 half-days. On the left, true T, U and S over 5 time steps.. In the middle, prediction of T, $U = (u, v)$ and S over 8 time steps. On the right, differences between targets and estimations.

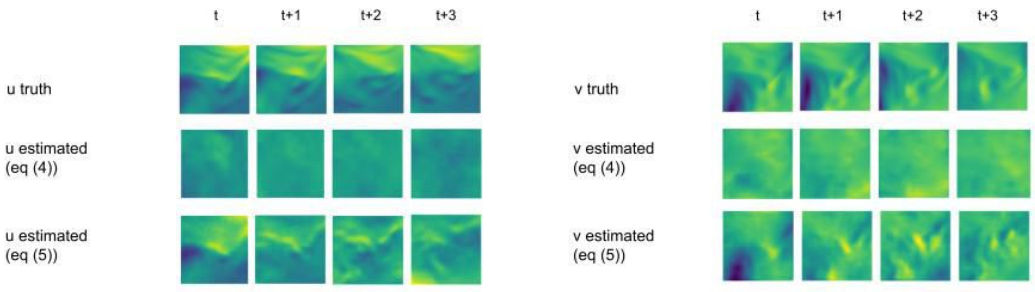

Figure 9: *Best viewed in color*. Sequence of estimations on $U = (u, v)$ for the Natl data. The second and third row respectively refer to training according to eq. (4) and eq. (5). The loss term $\mathrm{d}(h_k, f_k^{pr})$ in eq. (5) enables our model to learn more accurate velocity fields than when only trained following eq. (4).

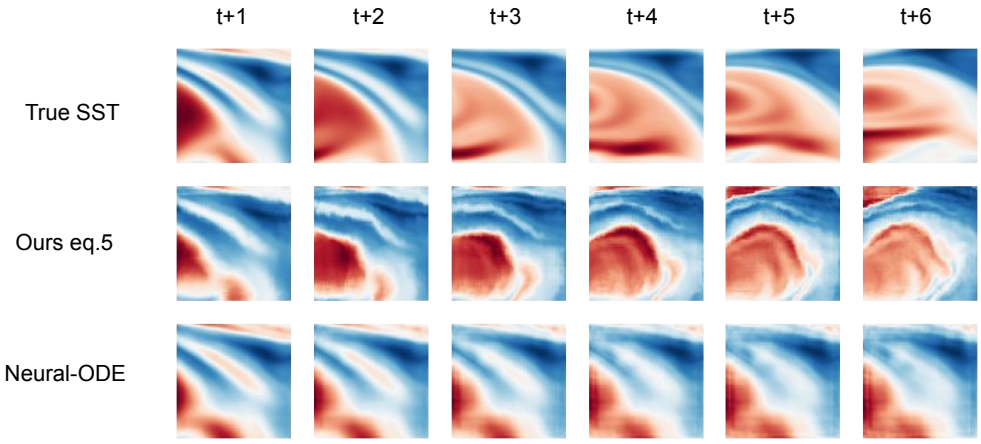

Figure 10: *Best viewed in color.* Sequence of prediction on T for the Natl data. Contrary to our model (row eq. (5)), NODE (row Neural-ODE) struggles to predict any motion in T.

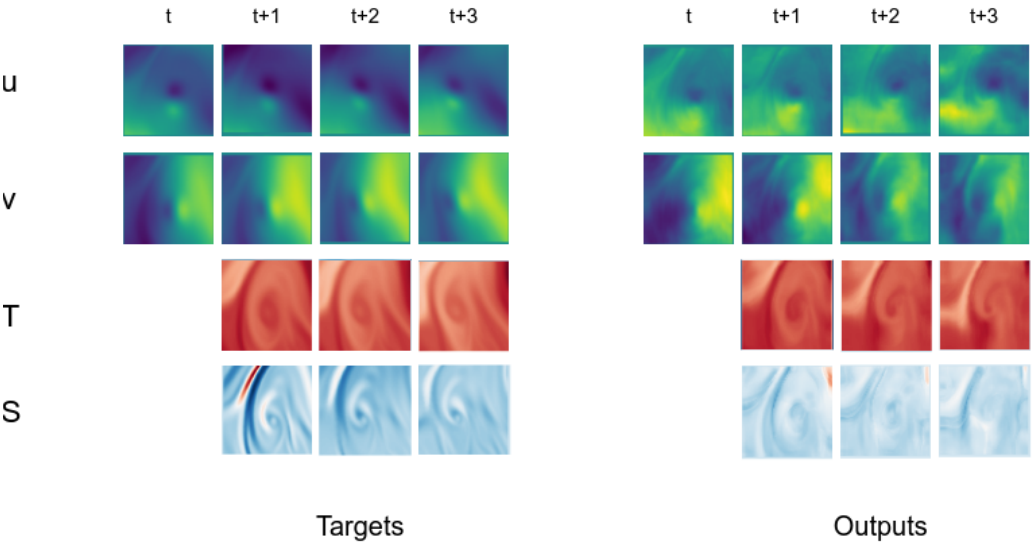

Figure 11: *Best viewed in color.* Sequence of prediction on T, u, v, S for the Natl data across 3 days trained using proxy data according to eq. (5)