VALÉRIAN JACQUES-DUMAS

LEARNING INVERSE AND FORWARD MODELS FROM PARTIALLY KNOWN DYNAMICS: APPLICATION TO OCEAN DYNAMICS

Encadrement : P. Gallinari (LIP6), M. Déchelle (LIP6), J. Dona (LIP6)

General description of the problem

Let $Z_t = (X_t, Y_t)$ be the complete of a dynamical system where X_t is observed and Y_t is not. This system obeys the following equation:

$$\frac{dZ_t}{dt} = \frac{d}{dt}\begin{pmatrix} X_t \\ Y_t \end{pmatrix} = \begin{pmatrix} f_X(Z_t) \\ f_Y(Z_t) \end{pmatrix}$$

The objective is to solve both the forward and inverse problem, that is to predict the upcoming values of X_t (forward problem) and to reconstruct Y_t and recover the full state Z_t (inverse problem).
But as nor Y_t neither F_Y is known, we focus on the equation describing the dynamics of X_t:

$$\frac{dX_t}{dt} = f_X(Z_t) = f_k(Z_t) + f_u(Z_t)$$

where f_k is fully known (but depends on unknown parameters) and corresponds to the physics model. f_u is not known (correction to the model) and should be learnt from the data.

Objective

Learn f_X from the data to recover the dynamics of X.
In practice, we aim at approximating f_X with a function $h = h_k + h_u$ learnt from the data.
h_k has the same form as f_k and depends on unknown parameters θ_k.
h_u is a free-form functional (as f_u is unknown) and considered here a neural network. It also depends on unknown parameters θ_u.

MB/ML (Model-Based / Machine Learning) Approach

Step 1: use knowledge of X_t and f_k (Model-Based approach) to estimate θ_k and Y_t (inverse problem)
Step 2: use knowledge of Y_t and θ_k to estimate θ_u from the data (Machine Learning approach) (forward problem)
The new knowledge of h_u is then used to improve the estimate of Y_t and θ_k when going back to step 1

Minimizing a distance

In practice, approximating f with h corresponds to minimizing the distance:

$$\mathrm{d}(h, f) = \mathbb{E}_{Z \sim p_Z} ||h(Z) - f(Z)||_2^2$$

with p_Z the distribution of Z.

Problem with this approach

As f is unknown and we have only acces to X, the distance d cannot be computed. Moreover, this minimization is an ill-posed problem: there may be an infinite number of combinations of the model-based and data-based dynamics.

Idea

We would like h_k to describe as much as possible of the dynamics and to be as close as possible to f_k. On the other hand, the contribution of h_u should be minimal.
Using the triangular inequality, we can derive upper bounds to the distance d. This would allow to control the contribution of each term in the solution.
The ill-posed problem is thus decomposed into well-posed sub-problems.

Direct constrain on both terms

In a first attempt, the following upper-bound can be derived:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h, h_k) + \mathrm{d}(h_k, f) \qquad (1)$$

where $\mathrm{d}(h, h_k) = ||h_u||_2^2$, which allows to constrain the contribution of h_u to be minimal. Moreover, we also constrain h_k to describe as much as possible of the dynamics. But on the other hand we are not assured that h_k will match f_k.

Upper bound using physical a priori

Ideally we would like to have a term $\mathrm{d}(h_k, f_k)$. Since f_k has unknown parameters, it is impossible. However, if we can access proxy data for f_k (denoted f_k^p and extracted from total dynamics $f^p = f_k^p + f_u^p$), then $\mathrm{d}(h_k, f_k^p)$ is tractable. The upper bound then becomes:

$$\mathrm{d}(h, f) \leq \mathrm{d}(h_k, f_k^p) + \mathrm{d}(h, h_k) + \mathrm{d}(f, f^p) + \mathrm{d}(f^p, f_k^p) \qquad (2)$$

The first term ensures the interpretability of h_k. The last two terms are constants that depend only on this proxy data and thus cannot be minimized.

Practical minimization

In practice, as we alternatively learn the parameters for h_k (associated distance $\ell(h_k) = \mathrm{d}(h_k, f)$ in the case of eq. (1) and $\ell(h_k) = \mathrm{d}(h_k, f_k^p)$ in the case of eq. (2)) and h_u (associated distance $\mathrm{d}(h, h_k)$, we need two loss functions to minimize. They respectively take the form:

$$\mathcal{L}_k(h_k) = \lambda_h \mathrm{d}(h, f) + \lambda_{h_k} \ell(h_k) \qquad \mathcal{L}_u(h_u) = \lambda_h \mathrm{d}(h, f) + \lambda_{h_u} \mathrm{d}(h, h_k)$$

Lagrangians are used to allow minimization of the distance related to h_k or h_u while imposing minimization of $\mathrm{d}(h, f)$ as additional constraint.

Application to Ocean Dynamics: basic equation

To apply this method, we study a system describing the dynamics of ocean surface variables: its surface temperature, denoted T, advected by the surface velocity, denoted U and subject to a forcing S

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

Here, only T can be observed. Thus, $X = T$ and $Y = U = (u, v)$.
The known term of the dynamics correspond to the advection $f_k = -\nabla.(TU)$. The source term S is the unknown term f_u of the dynamics. More precisely, S is a sum of two terms: a term describing the ocean-atmosphere heat exchanges (which we can access in the data) and a term describing the interactions with the inner ocean layers (which is unknown).

NATL Data

Data used in this work comes from NATL60: a high-resolution simulation emulating real world observations. We have access to daily data over a year spanning a 2300×2560km zone over the North-Atlantic Ocean, with a resolution of 60° (1.5km).

Proxy-data: using SSH to estimate T

If our data came from satellite observations, we would not have had access to U. But we would have access to SSH (Sea Surface Height), from which we can compute a proxy for U (equivalent of f_k^p here), under the assumption of slow movement:

$$v = \gamma \frac{\partial SSH}{\partial x}, \qquad u = -\gamma \frac{\partial SSH}{\partial y}$$

with γ a known constant.

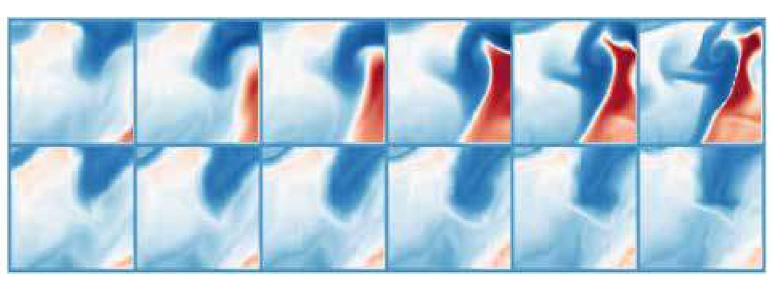
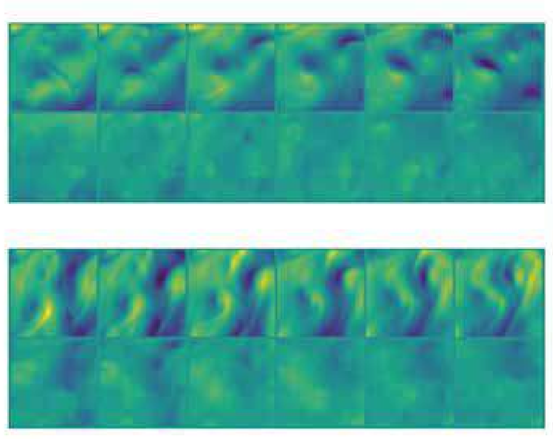


Figure: Top: 6 consecutive fields of T in the data. Bottom: prediction using the upper bound 1. The corresponding fields of U and S are shown below.



(a) Top of each line: 6 consecutive fields of U in the data. Bottomof each line: prediction using the upper bound 1

(b) Top: 6 consecutive fields of S in the data. Bottom: prediction using the upper bound 1

Evaluating Results

Two metrics are used: MSE (Mean Squared Error) and SSIM (Structural SIMilarity).
MSE measures the quality of predicted fields "pixel by pixel", the closer to 0 the better. If we have n ground-truth fields T and corresponding predictions $\tilde{T}$, the MSE is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(T_i - \tilde{T}_i)^2$$

SSIM measures the quality of the predicted images: contrast, structures... The closer to 1 the better.
All the results below are obtained after 30 epochs (going 30 times through the training dataset), for three ways of estimating f: minimizing $\mathrm{d}(h, f)$ only, and using each of the upper bound presented above.

Variables	T	U	S
Only $\mathrm{d}(h, f)$	8.09(1.63E-3)	11.67(8.42E-4)	6.05(1.76E-4)
Upper bound (1)	9.05(1.53E-3)	11.65(1.58E-3)	6.08(1.03E-3)
Upper bound (2)	9.53(3.56E-3)	7.77(2.35E-3)	6.07(4.01E-4)

Table: Mean MSE ($\times 100$) computed on 5 independent learnings with the corresponding standard deviation.

Variables	T	U	S
Only $\mathrm{d}(h, f)$	74.18(9.52E-3)	68.11(1.14E-2)	65.62(3.67E-2)
Upper bound (1)	71.56(5.18E-3)	65.71(3.96E-3)	68.10(3.80E-2)
Upper bound (2)	73.71(1.05E-2)	80.03(1.48E-3)	71.75(2.54E-3)

Table: Mean SSIM ($\times 100$) computed on 5 independent learnings with the corresponding standard deviation.