Environmental Data Science (2023), 2: e2, 1–10
doi:10.1017/eds.2022.31

APPLICATION PAPER

Deep prior in variational assimilation to estimate an ocean circulation without explicit regularization

Arthur Filoche[1,*], Dominique Béréziat[1] and Anastase Charantonis[2]

[1]LIP6, CNRS, Sorbonne Université, Paris 75005, France
[2]ENSIIE, CNRS, LAMME, France
*Corresponding author. E-mail: arthur.filoche@lip6.fr

Received: 27 October 2022; **Accepted:** 29 October 2022

Keywords: Deep prior; ocean circulation; regularization; twin experiment; variational data assimilation

Abstract

Many applications in geosciences require solving inverse problems to estimate the state of a physical system. Data assimilation provides a strong framework to do so when the system is partially observed and its underlying dynamics are known to some extent. In the variational flavor, it can be seen as an optimal control problem where initial conditions are the control parameters. Such problems are often ill-posed, regularization may be needed using explicit prior knowledge to enforce a satisfying solution. In this work, we propose to use a deep prior, a neural architecture that generates potential solutions and acts as implicit regularization. The architecture is trained in a fully-unsupervised manner using the variational data assimilation cost so that gradients are backpropagated through the dynamical model and then through the neural network. To demonstrate its use, we set a twin experiment using a shallow-water toy model, where we test various variational assimilation algorithms on an ocean-like circulation estimation.

Impact Statement

Data assimilation is the operational tool of choice when it comes to forecast Earth systems. The classical variational assimilation modeling framework is built on Gaussian prior hypothesis then relying on second-order statistics as hyperparameters, which may be hard to estimate. Inspired by deep image priors, we propose a hybrid method bridging a neural network and the 4D-Var mechanistic constraints to assimilate observation without specifying any statistics. It also constitutes a step toward bridging variational assimilation and deep learning, and extends the application domain of unlearned methods based on deep priors.

1. Introduction

Physics-driven numerical weather prediction requires estimating initial conditions before making a forecast. To do so, one should exploit all the knowledge at disposal which can be observations, a dynamical model, or errors statistics. It formalizes as an inverse problem and data assimilation offers a large panel of methods to solve it (Asch et al., 2016). A subset of these methods is variational so that the

This research article was awarded Open Data and Open Materials badges for transparent practices. See the Data Availability Statement for details.

system state is estimated via the minimization of a cost function as in the 4D-Var algorithm (Le Dimet and Talagrand, 1986). Many similarities with machine learning have been highlighted (Abarbanel et al., 2018) as both can be used to perform Bayesian inversion by gradient descent.

Even though variational data assimilation has a long-standing experience in model-constrained optimization, deep learning techniques have revolutionized ill-posed inverse problem solving (Ongie et al., 2020). And methods combining neural architectures and differentiable physical models already exist (Tompson et al., 2017; de Bezenac et al., 2018; Mosser et al., 2018). Most of the time, a large database is leveraged to learn a regularization adapted to the task.

Fitting observations respecting the dynamical model can be seen as a form of regularization but an additional regularizer may be required to promote an acceptable solution (Johnson et al., 2005). Recently a very original idea called "deep prior" (Ulyanov et al., 2018) has been developed. A neural architecture is used to generate the solution of an inverse problem and acts like an implicit regularization. Astonishingly the whole architecture is trained in an unsupervised manner on one example and provides results comparable to supervised methods.

In this work, we propose a hybrid methodology bridging deep prior and variational data assimilation and we test it in a twin experiment. The algorithm is evaluated on an ocean-like motion estimation task requiring regularization, then compared to adapted data assimilation algorithms (Béréziat and Herlin, 2014, 2018). All algorithms are implemented using tools from the deep learning community. The code is available on GitHub[1].

2. Methodology

2.1. Data assimilation framework

A dynamical system is considered where a state $\mathbf{X}$ evolves over time following perfectly-known dynamics $\mathbb{M}$, see equation (1). Partial and noisy observations $\mathbf{Y}$ are available through an observation operator $\mathbb{H}$, see equation (2). A background $\mathbf{X}_b$ gives prior information about the initial system state, see equation (3).

$$\text{Evolution}: \quad \mathbf{X}_{t+1} = \mathbb{M}_t(\mathbf{X}_t), \tag{1}$$

$$\text{Observation}: \quad \mathbf{Y}_t = \mathbb{H}_t(\mathbf{X}_t) + \varepsilon_{R_t}, \tag{2}$$

$$\text{Background}: \quad \mathbf{X}_0 = \mathbf{X}_B + \varepsilon_B. \tag{3}$$

Additive noise ε_B and ε_R represent uncertainties about the observations and the background, respectively. These noises are quantified by their assumed known covariance matrices $\mathbf{B}$ and $\mathbf{R}$, respectively. The dynamics is here considered perfect, but the framework could easily be extended to an imperfect dynamics. For any given matrix $\mathbf{A}$, we note $\|x-y\|_A^2 = \langle (x-y)|\mathbf{A}^{-1}(x-y)\rangle$ the associated Mahalanobis distance.

2.2. Variational assimilation

The objective of data assimilation is to provide an estimation of the system state $\mathbf{X}$ by optimally combining available data $\mathbf{X}_B$, $\mathbf{Y}$ and the dynamical model $\mathbb{M}$. In the variational formalism (Le Dimet and Talagrand, 1986), this is done via the minimization of a cost function which is the sum of background errors and observational errors, $\mathcal{J}_{\text{4DVar}} = \frac{1}{2}\|\varepsilon_b\|_{\mathbf{B}}^2 + \frac{1}{2}\sum_{t=0}^{T}\|\varepsilon_{R_t}\|_{\mathbf{R}_t}^2$. The optimization problem is model-constrained as described in equation (4). What motivates this cost function is that minimizing it leads to the maximum *a posteriori* estimation of the state under independent Gaussian

[1] https://github.com/ArFiloche/Deepprior4DVar_CI22.

errors, linear observation operator, and linear model hypothesis. The corresponding optimization algorithm is named as 4D-Var.

$$\begin{aligned} \underset{\mathbf{X}_0}{\arg\min} \quad & \mathcal{J}_{4\mathrm{DVar}}(\mathbf{X}_0) = \frac{1}{2}\|\mathbf{X}_0 - \mathbf{X}_b\|^2_{\mathbf{B}} + \frac{1}{2}\sum_{t=0}^{T}\|\mathbf{Y}_t - \mathbb{H}_t(\mathbf{X}_t)\|^2_{\mathbf{R}_t}, \\ \text{s.t.} \quad & \mathbf{X}_{t+1} = \mathbb{M}_t(\mathbf{X}_t). \end{aligned} \tag{4}$$

The link between variational assimilation and Tikhonov regularization is well described in Johnson et al. (2005). For example, choosing a particular matrix **B** will promote a particular set of solutions. Therefore, making alike choices can be seen as a handcrafted regularization to take advantage of expert prior knowledge.

2.3. Deep prior 4D-Var

The idea behind deep prior is that using a well-suited neural architecture to generate a solution of the variational problem can act as a handcrafted regularization. This means that the control parameters are shifted from the system state space to the neural network parameters space. From a practical standpoint, a latent variable z is fixed and a generator network g_θ outputs the solution from it such that $g_\theta(z) = \mathbf{X}_0$.

2.3.1. Cost function

The generator is then trained with the variational assimilation cost $\mathcal{J}(\theta)$. To emphasize the regularizing effect of the deep prior method, we choose to fix $\mathbf{B} = 0$ so that no background information is used. It means that $\mathcal{J}(\theta) = \frac{1}{2}\sum_{t=0}^{T}\|\varepsilon_{R_t}\|^2_{R_t}$ and by denoting multiple integration between two times $\mathbb{M}_{t_1\to t_2}$, the cost can be developed as in equation (5).

$$\mathcal{J}(\theta) = \frac{1}{2}\sum_{t=0}^{T}\|\mathbf{Y}_t - \mathbb{H}_t(\mathbb{M}_{0\to t}(g_\theta(z)))\|^2_{R_t} \tag{5}$$

It is important to note that this approach is unsupervised, the architecture being trained from scratch on one assimilation window with no pre-training. All the prior information should be contained in the architecture choice.

2.3.2. Gradient

The gradients of this cost function can be determined analytically. First, the chain rule gives equation (6). Then using the adjoint state method, we can develop $\nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0)$ as in equation (7), a detailed proof can be found in Asch et al. (2016). In the differentiable programming paradigm, such analytical expression is not needed to obtain gradients, adjoint modeling is implicitly performed as gradients are backpropagated automatically.

$$\nabla_\theta\mathcal{J}(\theta) = \nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0)\nabla_\theta\mathbf{X}_0 = \nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0)\nabla_\theta g_\theta(z) \tag{6}$$

$$\nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0) = \sum_{t=0}^{T}\left[\frac{\partial(\mathbb{H}_t\mathbb{M}_{0\to t})}{\partial\mathbf{X}_0}\right]^T \mathbf{R}_t^{-1}\varepsilon_{R_t} \tag{7}$$

2.3.3. Algorithm

The algorithm seeks to numerically optimize the cost function and simply consists of alternating forward and backward integration to update the control parameters by gradient descent (see Algorithm 1). A schematic view of the forward integration can be found in Figure 1.

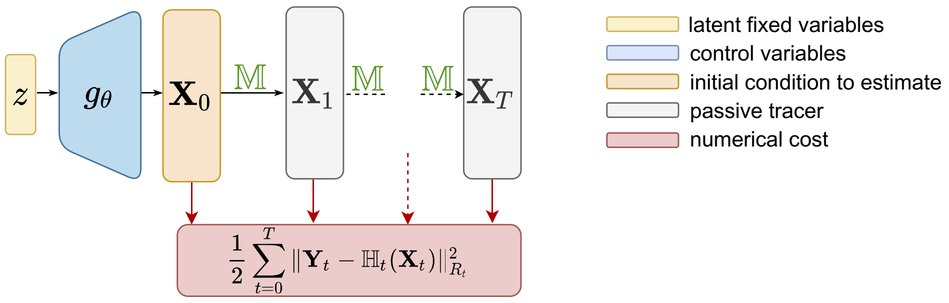

Figure 1. *Schematic view of the forward integration in deep prior 4D-Var.*

Algorithm 1 – Deep prior 4D-Var.

```
Initialize fixed latent variables $z$.
Initialize control variables $\theta$.
while stop criterion do.
    forward: integrate $\mathbb{M}_{0\to T}(g_\theta(z))$ and compute $\mathcal{J}$.
    backward: automatic differentiation returns $\nabla_\theta \mathcal{J}$.
    update: $\theta = \text{optimizer}(\theta, \mathcal{J}, \nabla_\theta \mathcal{J})$.
end while.
return $\theta, \mathbf{X}_0$.
```

3. Case Study

3.1. Twin experiment

The proposed methodology is tested within a twin experiment where data are generated from a numerical dynamical model. Observations are then created by sub-sampling and adding noise. The aim of this experiment is to highlight the implicit regularizing effect of the deep generative network. To do so, we compare various algorithms on the observation assimilation task. The considered algorithms are 4D-Var with no regularization, 4D-Var with Tikhonov regularization, and deep prior 4D-Var. All the algorithms are implemented with tools based on automatic differentiation such that no adjoint modeling is required, as described in Filoche et al. (2021). In all the assimilation experiments, the dynamical model is perfectly known.

3.2. Dynamical system

3.2.1. State

State variables of the considered system are η, the height deviation of the horizontal pressure surface from its mean height, and $\mathbf{w}$, the associated velocity field. $\mathbf{w}$ can be decomposed in u and v, the zonal and meridional velocity, respectively. At each time t, the system state is then $\mathbf{X}_t = \left(\eta_t\ \mathbf{w}_t^T\right)^T$. The considered temporal window has a fixed size.

3.2.2. Shallow water model

The dynamical model used here corresponds to a discretization of the shallow water equations system in equation (8) with first-order upwind numerical schemes. These schemes are implemented using a natively differentiable software. H represents the mean height of the horizontal pressure surface and g the acceleration due to gravity. After reaching an equilibrium starting from Gaussian random initial conditions, system trajectories are simulated as shown in Figure 2.

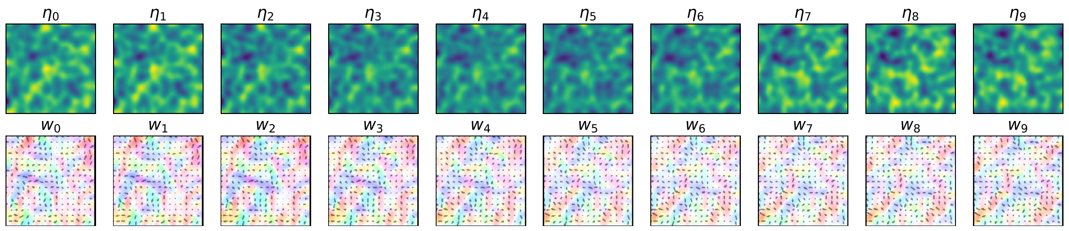

Figure 2. *Example of simulated trajectory with the shallow water numerical model.*

$$\begin{cases} \dfrac{\partial\eta}{\partial t}+\dfrac{\partial(\eta+H)u}{\partial x}+\dfrac{\partial(\eta+H)v}{\partial y}=0 \\ \dfrac{\partial u}{\partial t}+g\dfrac{\partial\eta}{\partial x}=0 \\ \dfrac{\partial v}{\partial t}+g\dfrac{\partial\eta}{\partial y}=0 \end{cases} \tag{8}$$

3.2.3. Observations

At regular observational dates, η is fully observed up to an additive white noise, see Figure 3. The velocity field $\mathbf{w}$ is never observed. This means that at observational date t, the observation operator $\mathbb{H}_t$ is then a linear projector so that $\eta_t = \mathbb{H}_t\mathbf{X}_t$.

3.3. Regularization

The role of the assimilation task is then to estimate the velocity field from successive observations of η. Such motion estimation inverse problem can be ill-posed and may need regularization. The dynamical model being considered perfect, all the velocity fields within the window are determined by the initial field $\mathbf{w}_0$.

3.3.1. No regularization

The 4D-Var version without regularization only optimizes the fit-to-data term in the cost function. This means that no prior knowledge on the solution can be used and the background covariance matrix $\mathbf{B}$ vanishes.

3.3.2. Tikhonov regularization

On the other hand, the "Tikhonov" 4D-Var algorithm optimizes the fit-to-data term and also a penalty term. The estimated motion field is forced to be smooth by constraining $\|\nabla\mathbf{w}_0\|_2^2$ and $\|\nabla.\mathbf{w}_0\|_2^2$ to be small. As proved in Lepoittevin and Herlin (2016), these terms can be directly included in the background error using a particular matrix $\mathbf{B}$ such that $\alpha\|\nabla\mathbf{w}_0\|_2^2+\beta\|\nabla.\mathbf{w}_0\|_2^2 = \|\mathbf{X}_0-\mathbf{X}_b\|_{B_{\alpha,\beta}}^2$, where α and β are the parameters to be tuned. Such regularization is a classical optical flow penalty (Horn and Schunck, 1981) and can be used for the sea-surface circulation estimation (Béréziat, 2000, Béréziat and Herlin, 2014).

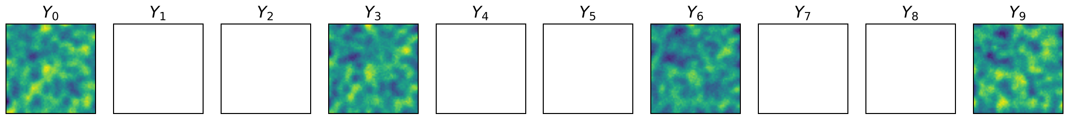

Figure 3. *Example of simulated system observations.*

3.3.3. Deep prior

As depicted in the method, the only assumption made is about the architecture of the network g_θ generating the solution $\mathbf{w}_0$. Obviously, the chosen architecture is critical for performance. In this experiment, we use the generative convolutional architecture introduced by Radford et al. (2016), but replacing deconvolution operations to avoid checkerboard artifacts as described in Odena et al. (2016). The exact architecture is provided in Appendix A.1, Figure 6.

3.3.4. Hyperparameters tuning

Regarding 4D-Var, α and β are tuned using Bayesian optimization on observations forecasts so that ground truth is never used. Finding hyperparameters, and particularly the number of epochs for DIP is still an active research field as described in Wang et al. (2021). Investigated early-stopping methods are beyond the scope of our study so we made the choice to fix the number of epochs.

4. Results

The first result to look at is the plot of the velocity fields estimated by the algorithms (Figure 4, 7). The 2D field of arrows represents the direction and the intensity of the velocity, the colormap provides the same information but helps visualization. Without regularization, the 4D-Var estimation is sharp and seems to suffer from numerical optimization artifacts. On the contrary, the deep prior estimate looks less precise but far smoother. The regularized provides the most accurate and smooth estimation. Other examples can be found in Appendix A.2.

Several metrics are calculated to quantify the quality of the estimations. The endpoint error $\|\widehat{\mathbf{w}}_0 - \mathbf{w}_0\|_2$ and the angular error $\arccos(\widehat{\mathbf{w}}_0, \mathbf{w}_0)$ are classical optical flow scores, they calculate the Euclidean distance and the average angular deviation between the estimation and the ground truth, respectively. At first glance, there is no statistical difference between deep prior 4D-Var and 4D-Var without regularization (Table 1).

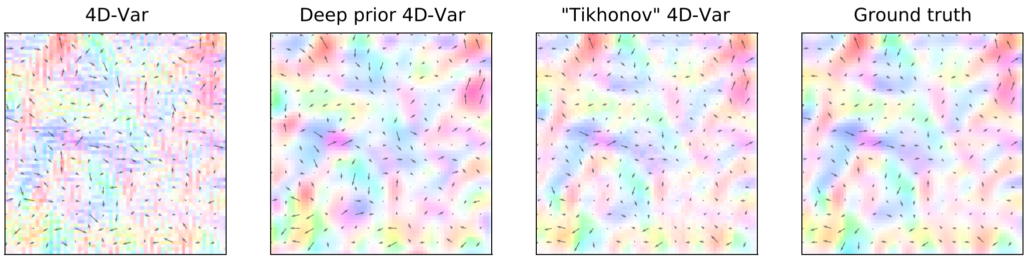

Figure 4. *Example of the estimated motion fields* $\mathbf{w}_0$ *with various algorithms.*

Table 1. Metrics quantifying the quality of the estimated motion field $\mathbf{w}_0$ over the assimilated database.

	Assimilation score		Smoothness statistics		
Metric[a]	Endpoint error$(\times 10^2)$	Angular error	$\|\nabla \mathbf{w}_0\|_2$	$\|\nabla . \mathbf{w}_0\|_2$	$\|\Delta \mathbf{w}_0\|_2$
4D-Var	04.2 ± 0.4	028.4 ± 9.8	06.1 ± 0.6	05.3 ± 0.5	09.9 ± 1.0
Deep prior 4D-Var	04.6 ± 2.0	026.7 ± 5.0	01.9 ± 0.1	01.6 ± 0.9	01.0 ± 0.3
"Tikhonov" 4D-Var	01.6 ± 0.6	09.9 ± 9.8	02.0 ± 0.1	01.8 ± 0.1	01.9 ± 0.1
Ground truth	00	00	01.7 ± 0.9	01.6 ± 0.1	00.7 ± 0.3

[a]All the metrics are averaged on images, $\pm$ precising standard deviation.

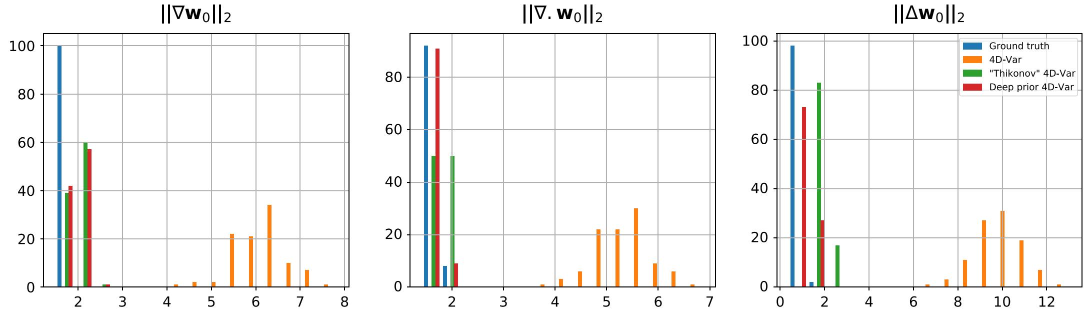

Figure 5. *Histograms of smoothness statistics from the estimated motion field* $\mathbf{w}_0$ *with various algorithms.*

However, if we dig into the smoothness statistics of the estimated fields, $||\nabla.\mathbf{w}_0||_2$, $||\nabla.\mathbf{w}_0||_2$, and $||\Delta\mathbf{w}_0||_2$, it seems that deep prior is able to capture complex statistics of the true motion field. Similar behavior has been noticed in Yoo et al. (2021)). Looking closer at the histograms, Figure 5, we see that deep prior and "Tikhonov" 4D-Var estimations have smoothness statistics very close to that of the original motion field. Whether it has been explicitly constrained or by deep network design, smooth solutions are unforced by regularization.

It has to be noted that the "Tikhonov" 4D-Var is the only algorithm here that has been treated with hyperparameters tuning which can explain the large differences in scores. It can be argued that the neural architecture, which is known as a good baseline to generate images, has been tuned through many image processing experiments. However, grid-searching hyperparameters particularly suited for this experiment should enhance performances.

5. Conclusion

We proposed an original method bridging ideas from the image processing and the geosciences communities to solve a variational inverse problem. More precisely, we used a neural network as implicit regularization to generate the solution of an initial value problem. To demonstrate its efficiency, we set up a twin experiment comparing different algorithms in a data assimilation task derived from the shallow water model. The results show that this kind of regularization can provide an interesting alternative when prior knowledge is not available. However, in our case, we observed that expert-driven handcrafted regularization provides better performances. Finally, this work opens the way for further developments on the architecture design, but also in a more realistic context where the numerical dynamics is imperfect.

Author Contributions. Conceptualization: all authors; Data visualization: A.F.; Methodology: A.F.; Project administration: D.B., A.C.; Writing—original draft: A.F.; Writing—review and editing: all authors. All authors approved the final submitted draft.

Competing Interests. The authors declare no competing interests exist.

Data Availability Statement. Data are synthetic, all experiments are reproducible through the GitHub repository provided by the authors.

Ethics Statement. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

Funding Statement. This research was supported by Sorbonne Center for Artificial Intelligence.

Provenance. This article is part of the Climate Informatics 2022 proceedings and was accepted in *Environmental Data Science* on the basis of the Climate Informatics peer review process.

References

Abarbanel H, **Rozdeba P and Shirman S** (2018) Machine learning: Deepest learning as statistical data assimilation problems. *Neural Computation 30*(8), 2025–2055.

Asch M, **Bocquet M and Nodet M.** (2016) *Data Assimilation: Methods, Algorithms, and Applications. Fundamentals of Algorithms*. SIAM. 978-1-61197-453-9.
Béréziat D and Herlin I (2014) Image-based modelling of ocean surface circulation from satellite acquisitions. In *International Conference on Computer Vision Theory and Application (VISAPP)*, Lisbon, 1–8, January 2014.
Béréziat D and Herlin I (2018) Motion and acceleration from image assimilation with evolution models. *Digital Signal Processing 83*, 45–58.
Bereziat D, **Herlin I and Younes L** (2000) A generalized optical flow constraint and its physical interpretation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, South Carolina, 13–15 June, volume *2*, 487–492.
de Bezenac E, **Pajot A and Gallinari P** (2018) Deep learning for physical processes: Incorporating prior scientific knowledge. In *International Conference on Learning Representations (ICLR)*, April, Vancouver.
Filoche A, **Béréziat D**, **Brajard J and Charantonis A** (2021) Variational assimilation of geophysical images leveraging deep learning tools. In *ORASIS 2021*, September 2021, Saint-Ferréol.
Horn B and Schunck B (1981) Determining optical flow. *Artificial Intelligence 17*, 185–203.
Johnson C, **Hoskins B and Nichols N** (2005) A singular vector perspective of 4D-Var: Filtering and interpolation. *Quarterly Journal of the Royal Meteorological Society 131*, 1–19.
Le Dimet F-X and Talagrand O (1986) Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects. *Tellus 38A*(10), 97.
Lepoittevin Y and Herlin I (2016) Regularization terms for motion estimation. Links with spatial correlations. *International Conference on Computer Vision Theory and Applications (VISAPP) 3*, 458–466.
Mosser L, **Dubrule O and Blunt M** (2018) Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. *Math Geosci 52*, 53–79 (2020). https://doi.org/10.1007/s11004-019-09832-6
Odena A, **Dumoulin V and Olah C** (2016) Deconvolution and checkerboard artifacts. *Distill*, 2016. http://doi.org/10.23915/distill.00003
Ongie G, **Jalal A**, **Metzler C**, **Baraniuk R**, **Dimakis A and Willett R** (2020) Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory 1*(1), 39–56.
Radford A, **Metz L and Chintala S** (2016) *Unsupervised representation learning with deep convolutional generative adversarial networks*. 4th International Conference on Learning Representations (ICLR), 2016, San Juan, Puerto Rico, May 2–4.
Tompson J, **Schlachter K**, **Sprechmann P and Perlin K** (2017) Accelerating Eulerian fluid simulation with convolutional networks. In *International Conference on Machine Learning (ICML)*, 5258–5267. August, Sidney.
Ulyanov D, **Vedaldi A and Lempitsky V** (2018) Deep image prior. In *Conference on Vision and Pattern Recognition (CVPR)*. June, Salt Lake City.
Wang H, **Li T**, **Zhuang Z**, **Chen T**, **Liang H and Sun J** (2021) Early stopping for deep image prior. In *CoRR*.
Yoo J, **Jin K**, **Gupta H**, **Yerly J**, **Stuber M and Unser M** (2021) Time-dependent deep image prior for dynamic MRI. *IEEE Transactions on Medical Imaging*, 2021, December, *40*(12'):3337–3348.

A. Appendix. Experiment details

A.1. Neural network architecture

Layer (type)	Output shape	Param #
ConvTranspose2d-1	[−1, 512, 4, 4]	819,200
BatchNorm2d-2	[−1, 512, 4, 4]	1,024
ReLU-3	[−1, 512, 4, 4]	0
Upsample-4	[−1, 512, 8, 8]	0
ReflectionPad2d-5	[−1, 512, 10, 10]	0
Conv2d-6	[−1, 256, 8, 8]	1,179,904
BatchNorm2d-7	[−1, 256, 8, 8]	512
ReLU-8	[−1, 256, 8, 8]	0
Upsample-9	[−1, 256, 16, 16]	0
ReflectionPad2d-10	[−1, 256, 18, 18]	0
Conv2d-11	[−1, 128, 16, 16]	295,040

Layer (type)	Output shape	Param #
BatchNorm2d-12	[−1, 128, 16, 16]	256
ReLU-13	[−1, 128, 16, 16]	0
Upsample-14	[−1, 128, 32, 32]	0
ReflectionPad2d-15	[−1, 128, 34, 34]	0
Conv2d-16	[−1, 64, 32, 32]	73,792
BatchNorm2d-17	[−1, 64, 32, 32]	128
ReLU-18	[−1, 64, 32, 32]	0
Upsample-19	[−1, 64, 64, 64]	0
ReflectionPad2d-20	[−1, 64, 66, 66]	0
Conv2d-21	[−1, 3, 64, 64]	1,731
Tanh-22	[−1, 3, 64, 64]	0

Trainable params: 2,371,587.

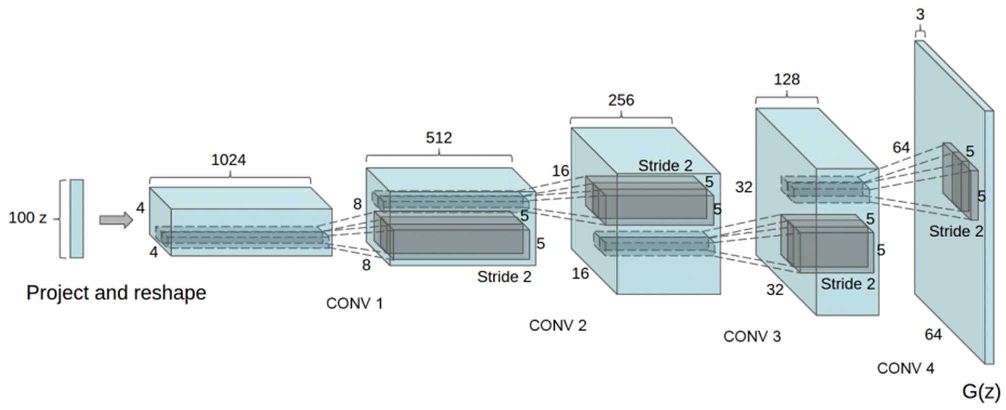

Figure 6. *The convolutional generator architecture from Radford et al. (2016).*

A.2. Supplementary examples

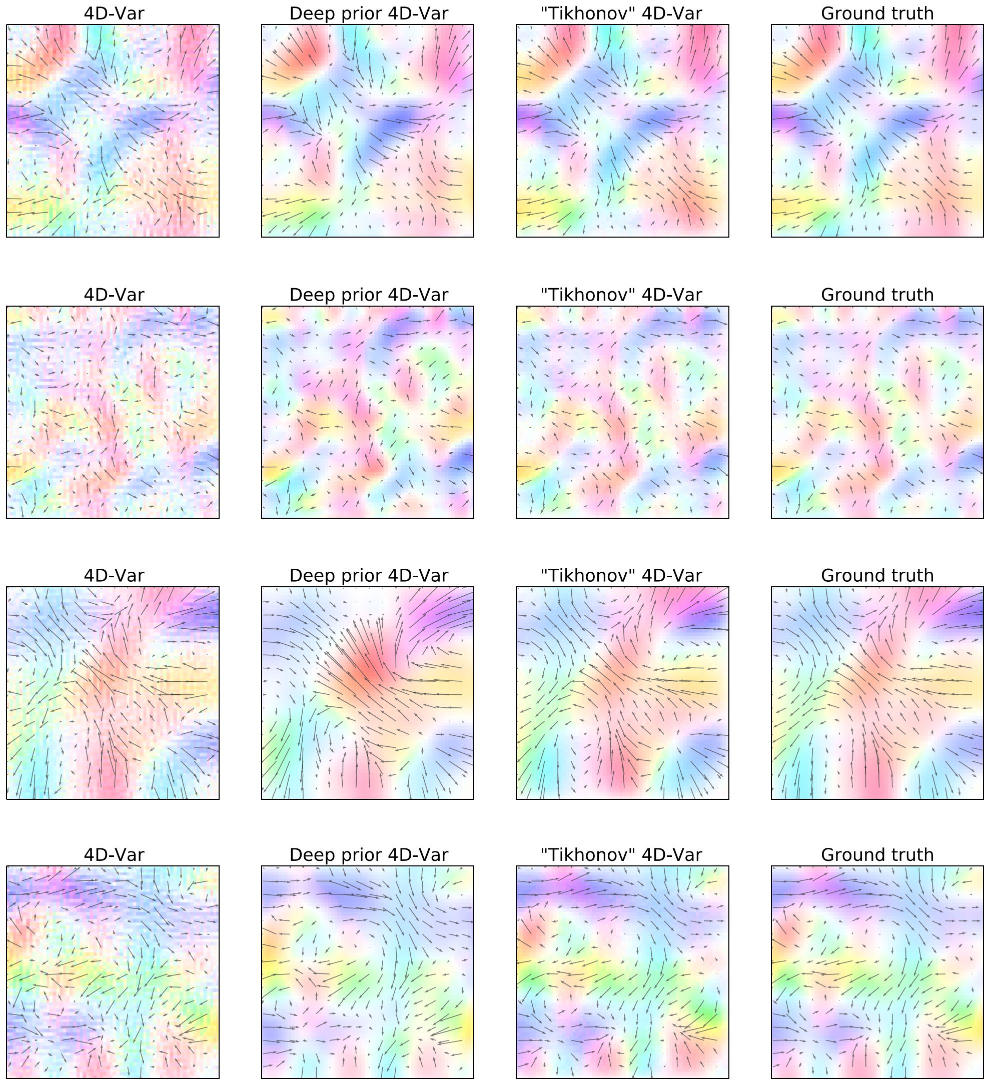

Figure 7. *Examples of the estimated motion fields* $\mathbf{w}_0$ *with various algorithms, each line corresponds to a different assimilation window.*