Lecture #5: Deep Learning and Inverse Problems in Geoscience

R. Fablet et al.
ronan.fablet@imt-atlantique.fr
web: cia-oceanix.github.io

Advanced course on Deep Learning and Geophysical Dynamics

Inverse Problems in Geoscience

Mathematical formulation for inverse Problems

Inverse problems & Deep learning

Applications to geophysical dynamics

Inverse Problems in Geoscience

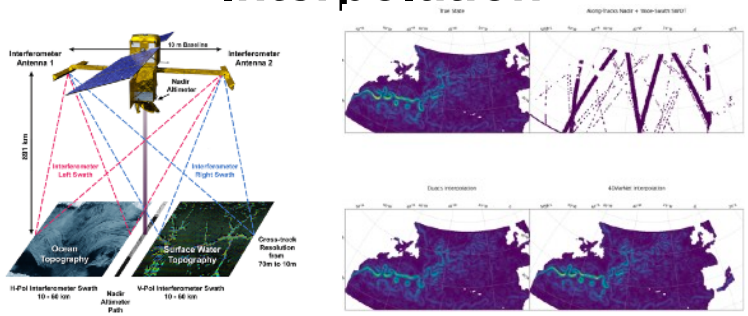
Mathematical formulations for inverse Problems

Inverse problems as learning problems

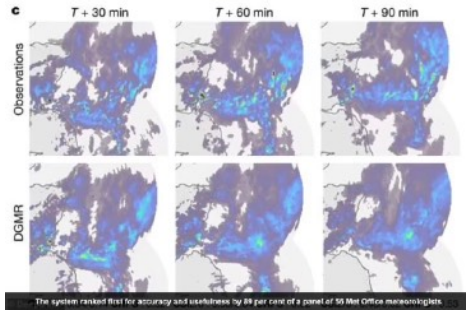
Applications to geophysical dynamics

Inverse Problems in Geoscience: some examples

Interpolation

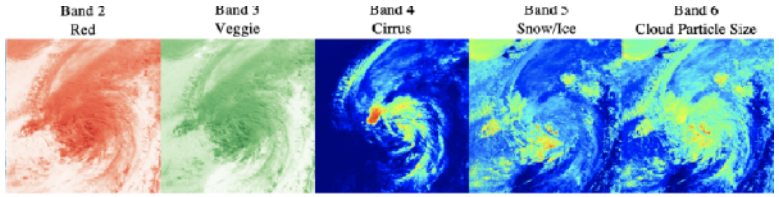


Obs.-driven Forecasting



Deepmind

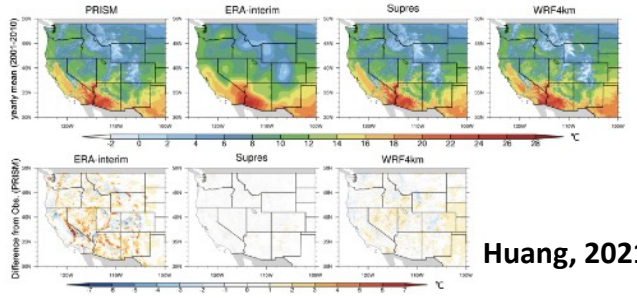
Multimodal fusion



Vandal et al.

Deconvolution

Downscaling

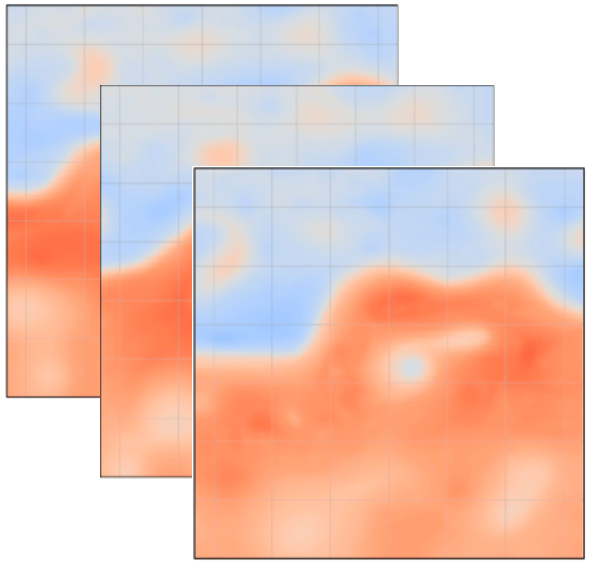


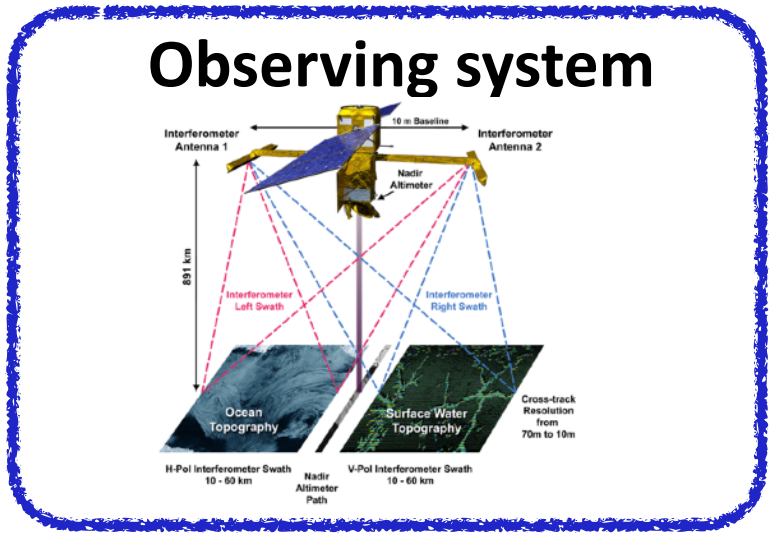
Huang, 2021

....

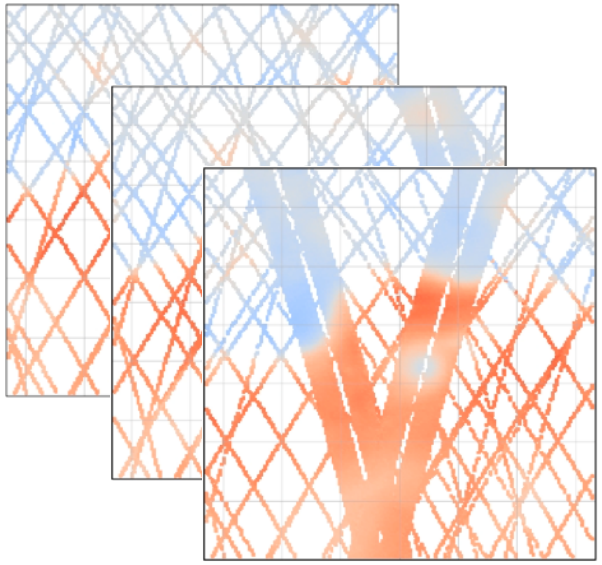
Inverse Problems in Geoscience: Generic formulation

State



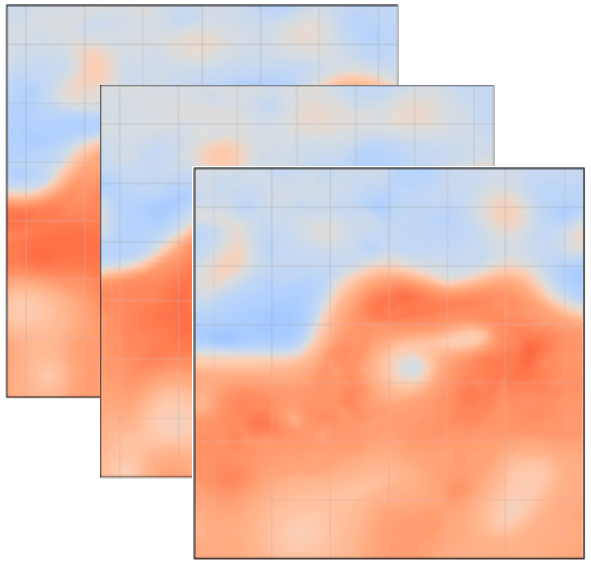


Observation



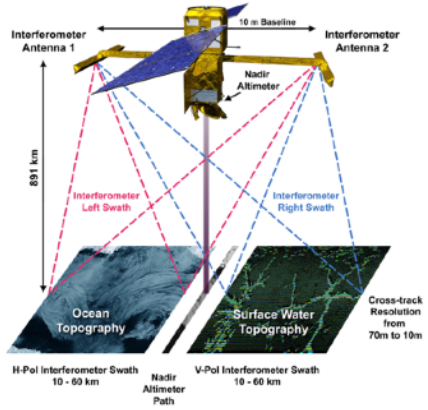
Inverse Problems in Geoscience: Generic formulation

State



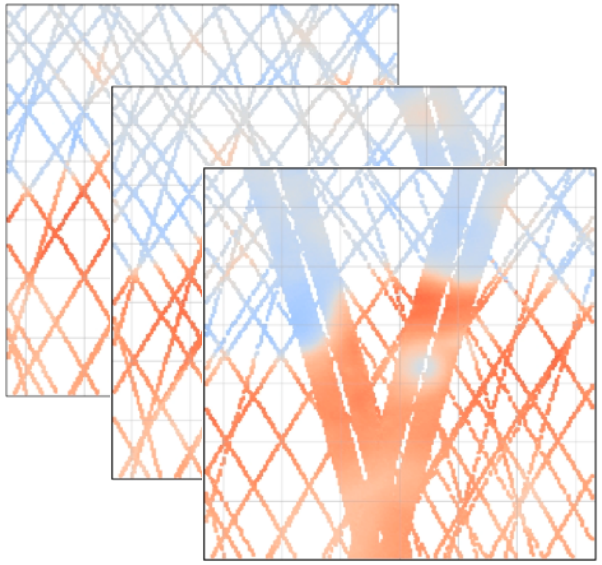
X

Observing system



Forward model F()

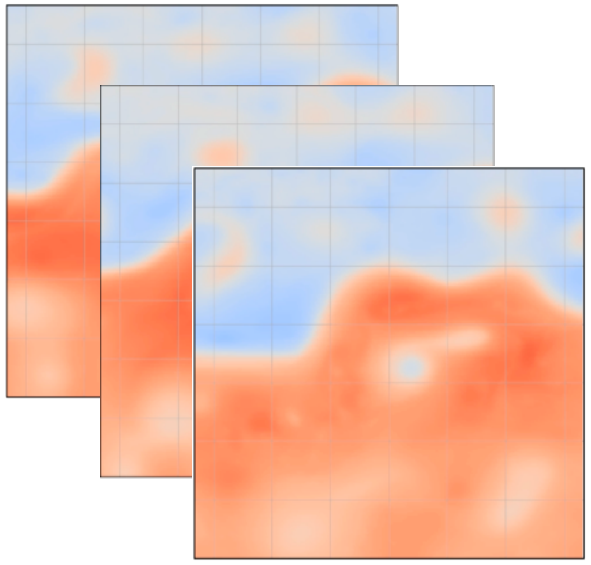
Observation



Y

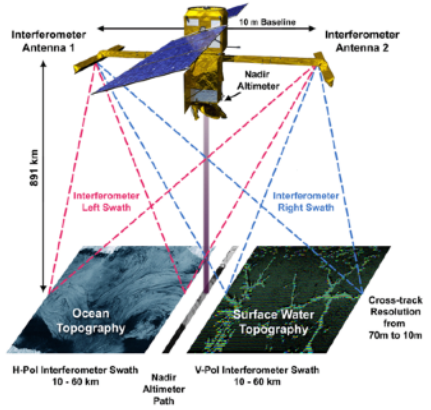
Inverse Problems in Geoscience: Generic formulation

State



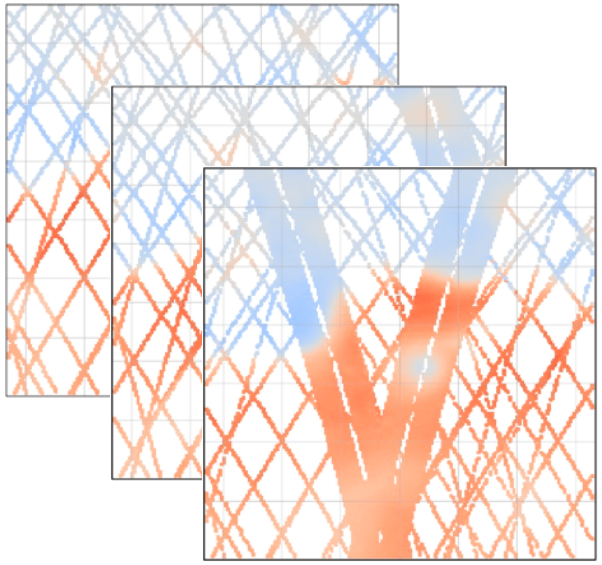
X

Observing system



Forward model H()

Observation

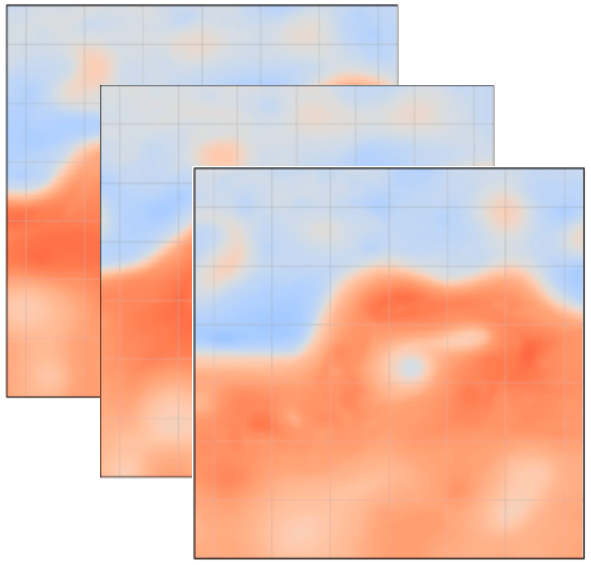


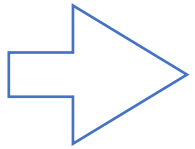
Y

Inverse model $H^{-1}()$

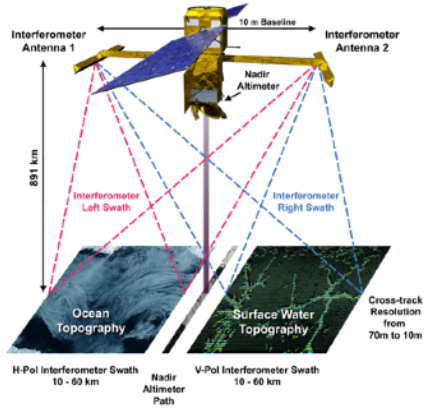
Inverse Problems in Geoscience: Examples of forward model

State

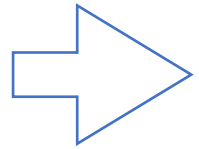




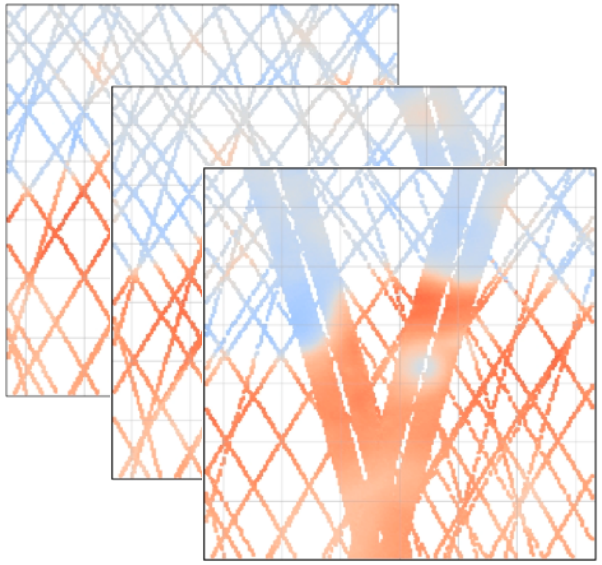
Observing system



Forward model H()

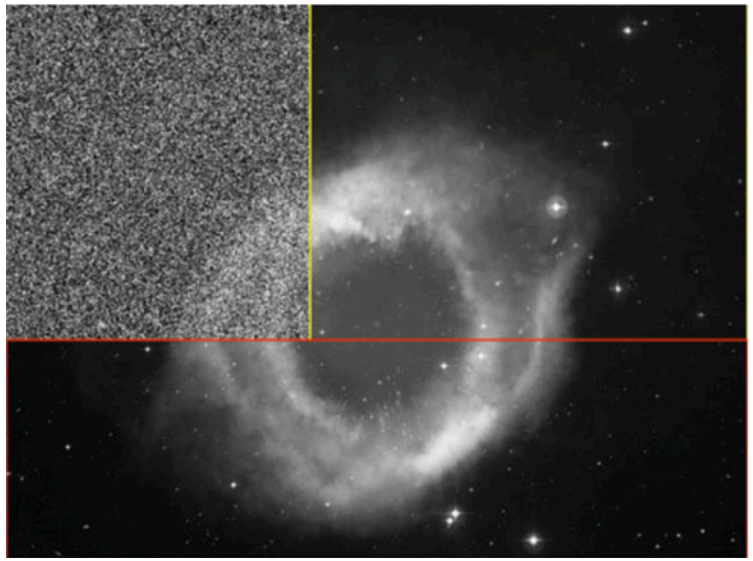


Observation

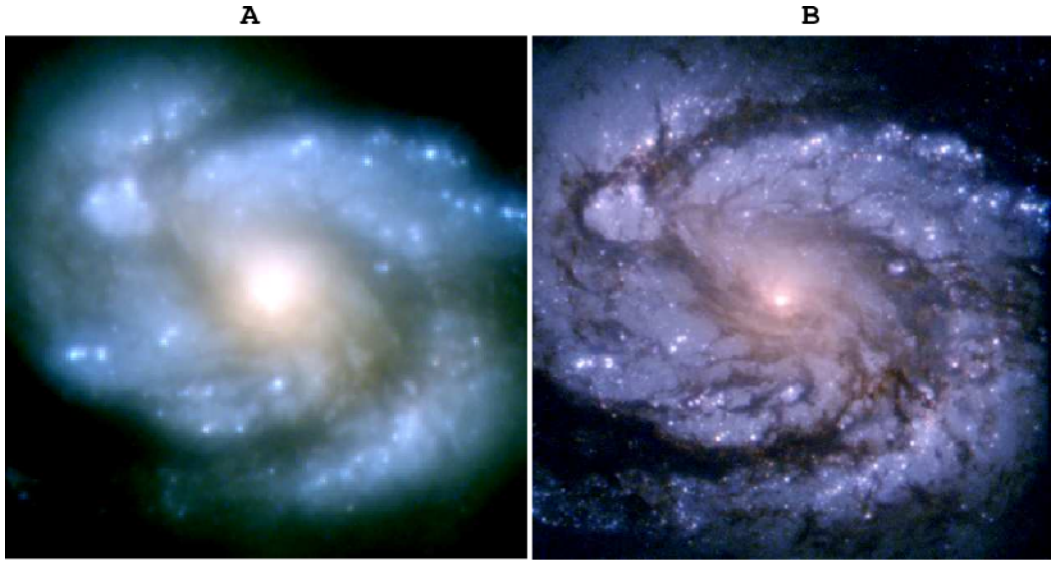


Inverse Problems in Geoscience: Examples of forward model

Denoising problem

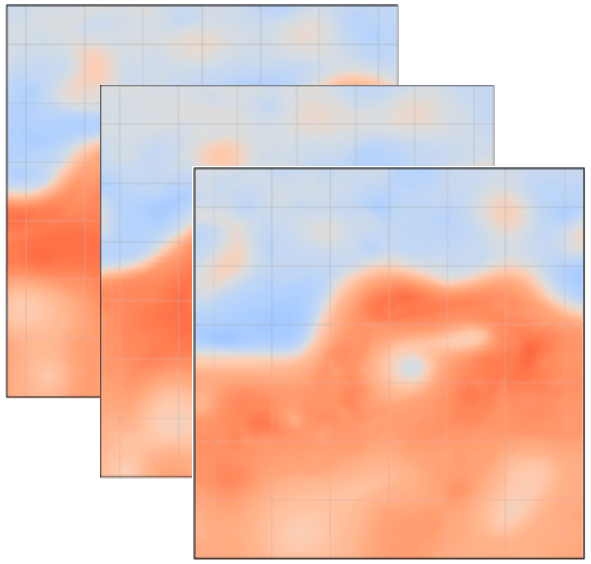


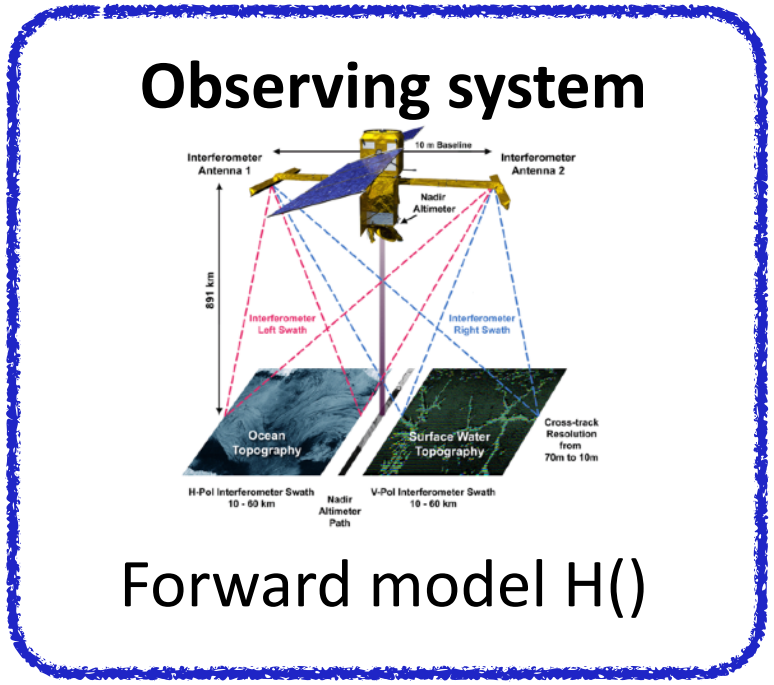
Downscaling problem



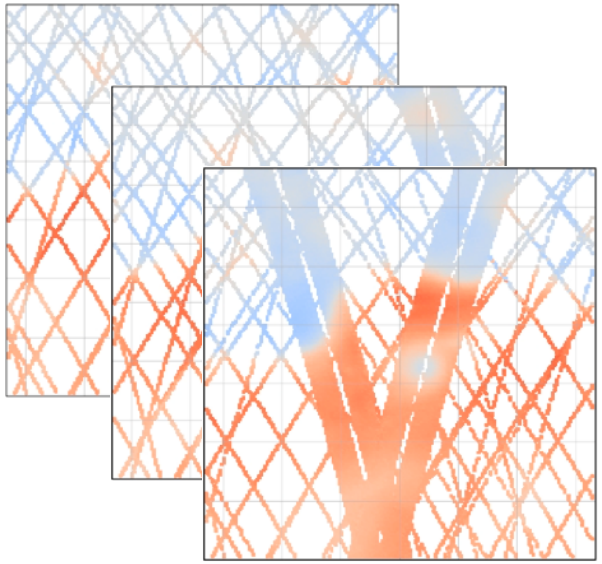
Inverse Problems and ill-posedness

State





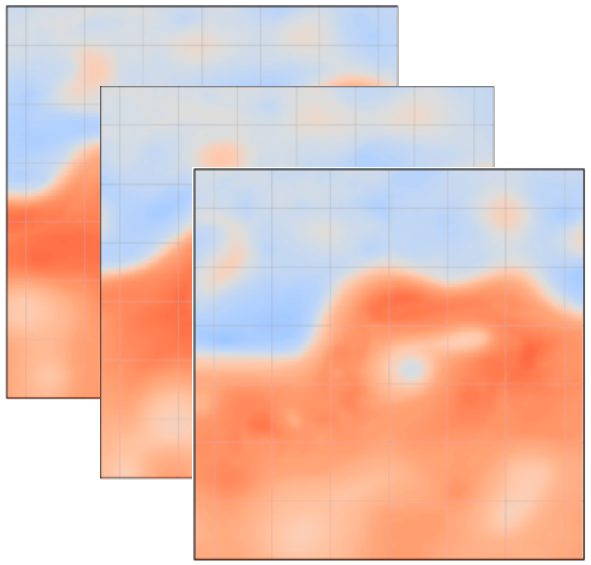
Observation



Why is space-time interpolation an ill-posed problem ?

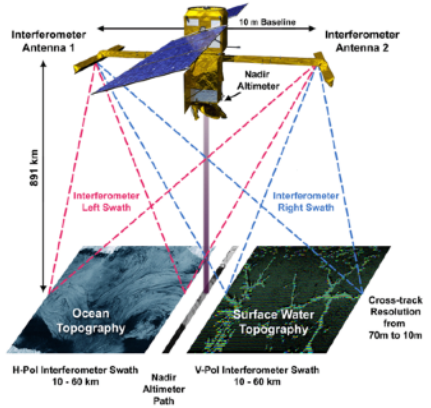
Inverse Problems and ill-posedness

State



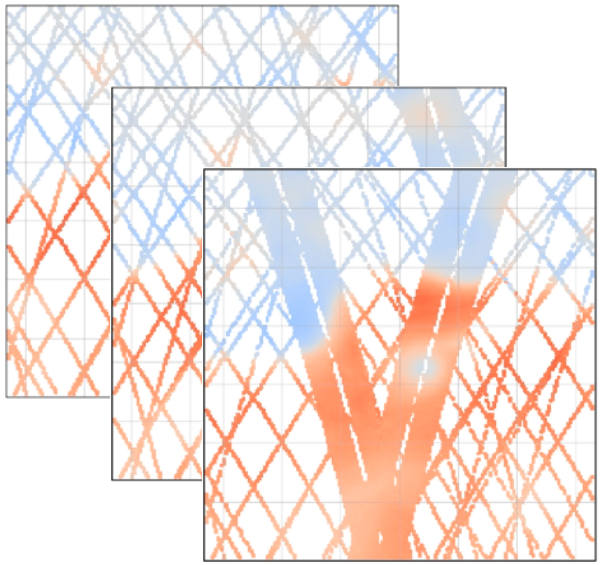
X

Observing system



Forward model H()

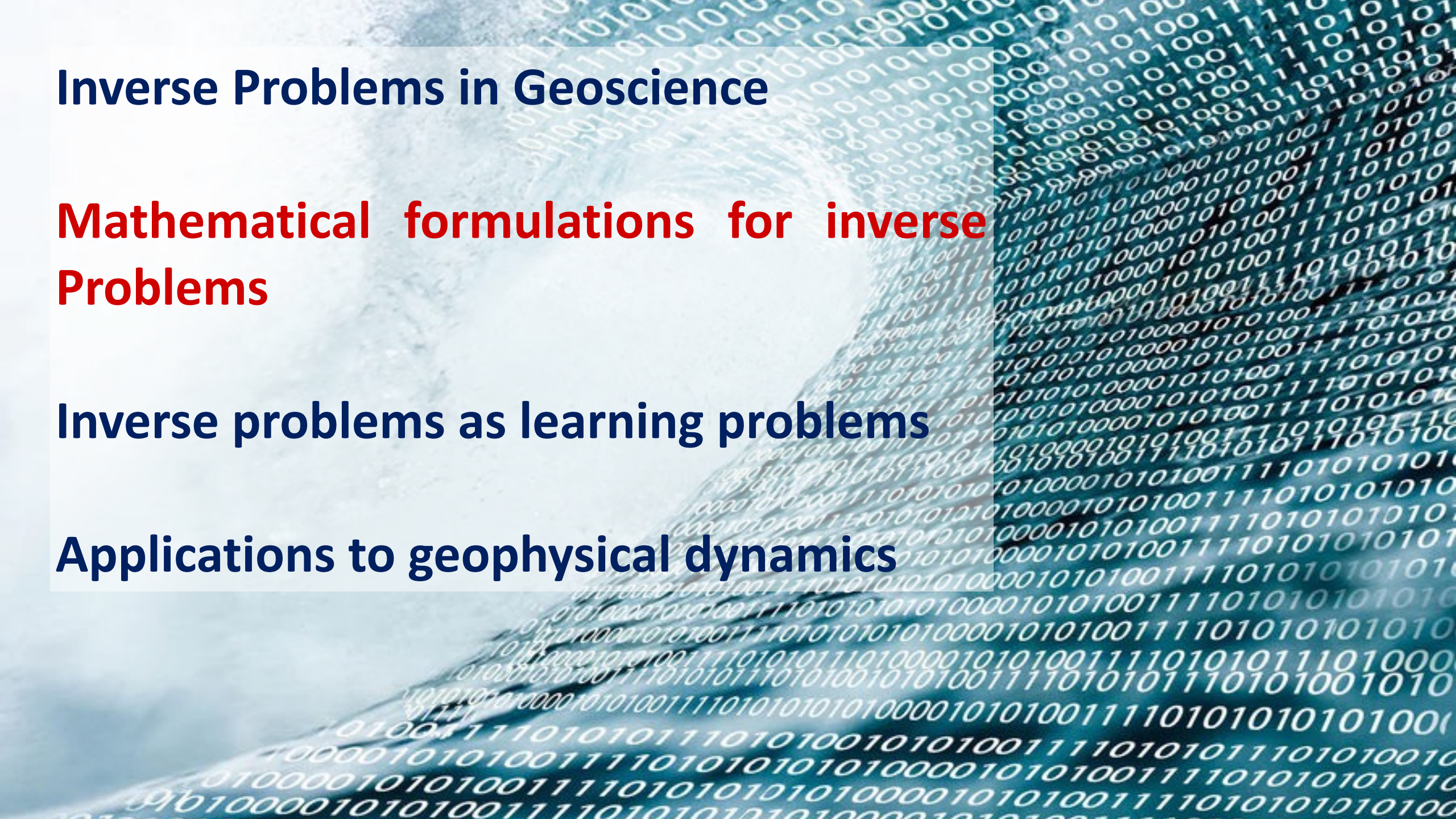
Observation



Y

Inverse model $H^{-1}()$

We need some additional information (prior) to make the problem invertible



Inverse Problems in Geoscience

Mathematical formulations for inverse Problems

Inverse problems as learning problems

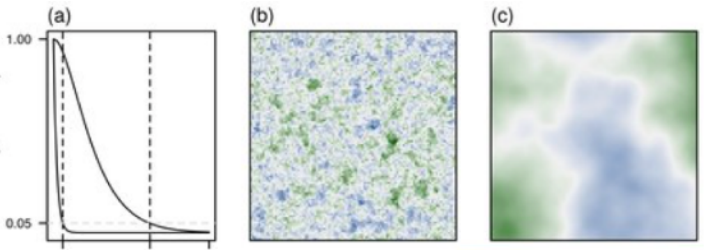
Applications to geophysical dynamics

Which priors to solve inverse problems?

Probablistic prior

$$X \sim P_X$$

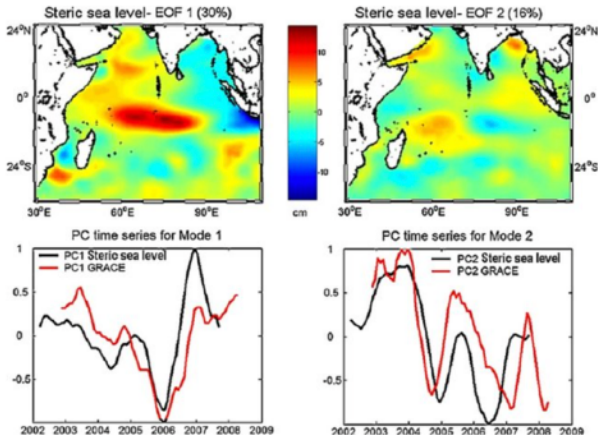
Gaussian process



Dictionary-based prior

$$X = \sum_k \alpha_k D_k$$

$$X = D.\alpha$$



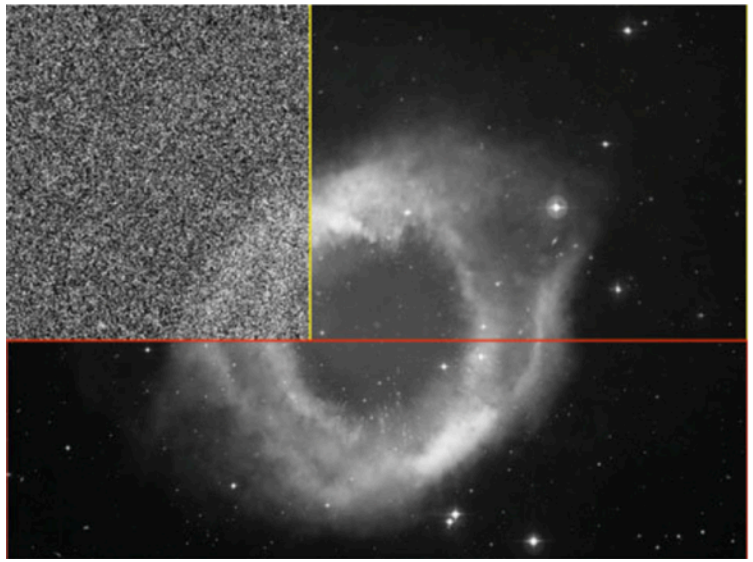
Norm-based prior

$$\|\nabla X\|_2^2$$

$$\|\nabla X\|$$

Inverse problems stated as minimisation problems

Denoising problem



$$y = x + \epsilon \text{ with } \epsilon \sim \mathcal{N}(0, \sigma^2)$$

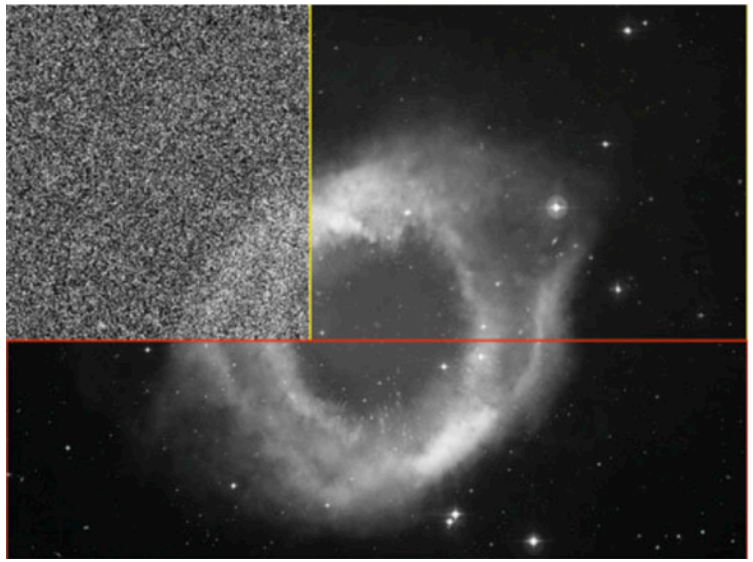
$$x \sim P_x$$

Probablistic prior

$$\widehat{x} = \arg\min_x \lambda \|y - x\|^2 - \log P_x(x)$$

Inverse problems stated as minimisation problems

Denoising problem



$$y = x + \epsilon \text{ with } \epsilon \sim \mathcal{N}(0, \sigma^2)$$

Probablistic prior

$$x \sim P_x$$

$$\widehat{x} = \arg\min_x \lambda \|y - x\|^2 - \log P_x(x)$$

Dictionary-based prior

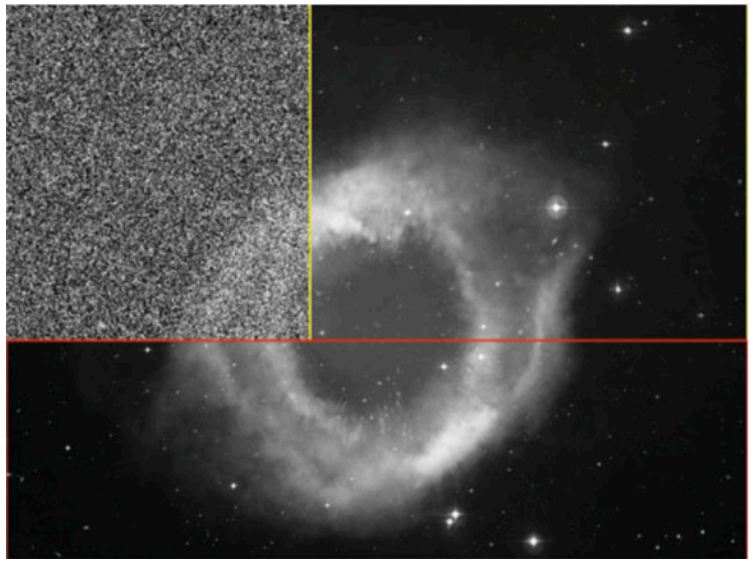
$$x = D.\alpha$$

$$\widehat{\alpha} = \arg\min_\alpha \|y - D.\alpha\|^2$$

Inverse problems stated as minimisation problems

Denoising problem

$$Y = X + \epsilon \text{ with } \epsilon \sim \mathcal{N}(0, \sigma^2)$$



Probablistic prior

$$X \sim P_X$$

$$X = \arg\min_X \lambda \|X - Y\|^2 - log P_X(X)$$

Dictionary-based prior

$$X = D.\alpha$$

$$X = \arg\min_\alpha \|Y - D.\alpha\|^2$$

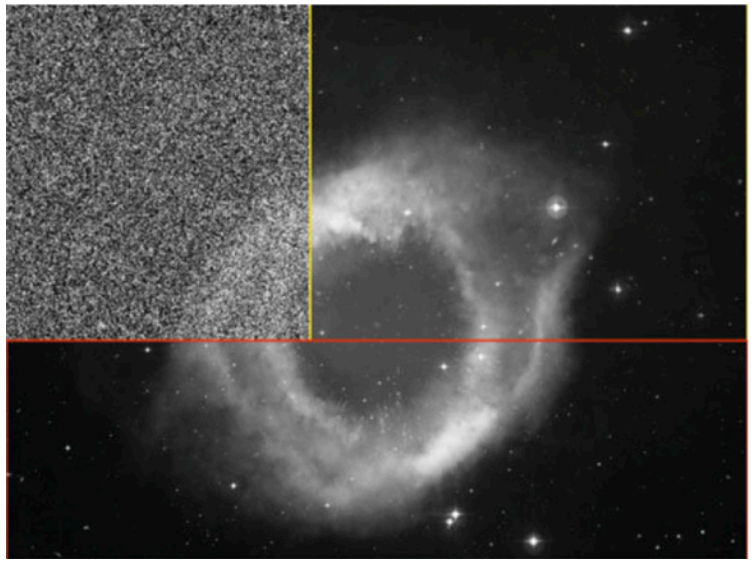
Norm-based prior

$$\widehat{x} = \arg\min_x \|y - x\|^2 + \lambda \|\nabla x\|^2$$

Inverse problems stated as minimisation problems

Denoising problem

$$Y = X + \epsilon \text{ with } \epsilon \sim \mathcal{N}(0, \sigma^2)$$



Probablistic prior

$$X \sim P_X$$

$$X = \arg\min_X \lambda \|X - Y\|^2 - log P_X(X)$$

Dictionary-based prior

$$X = D.\alpha$$

$$X = \arg\min_\alpha \|Y - D.\alpha\|^2$$

Norm-based prior

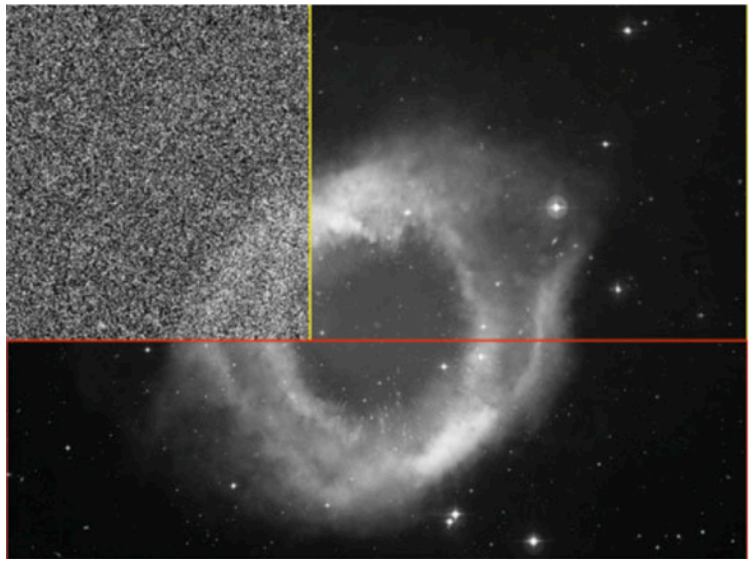
$$\widehat{x} = \arg\min_x \|y - x\|^2 + \lambda \|\nabla x\|^2$$

Generic formulation

$$\widehat{x} = \arg\min_x \|y - H(x)\|^2 + \lambda U_{Reg}(x)$$

Inverse problems stated as minimisation problems

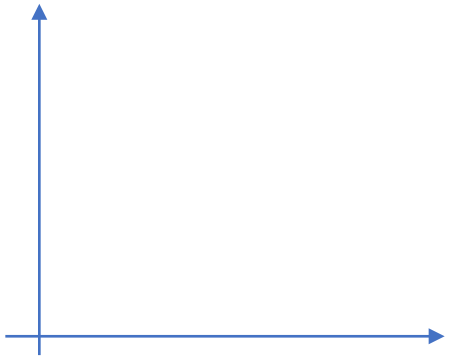
Denoising problem



Minimisation problem

$$X = \arg\min_X \|Y - H(X)\|^2 + \lambda U_{reg}(X)$$

How to solve the minimisation ?

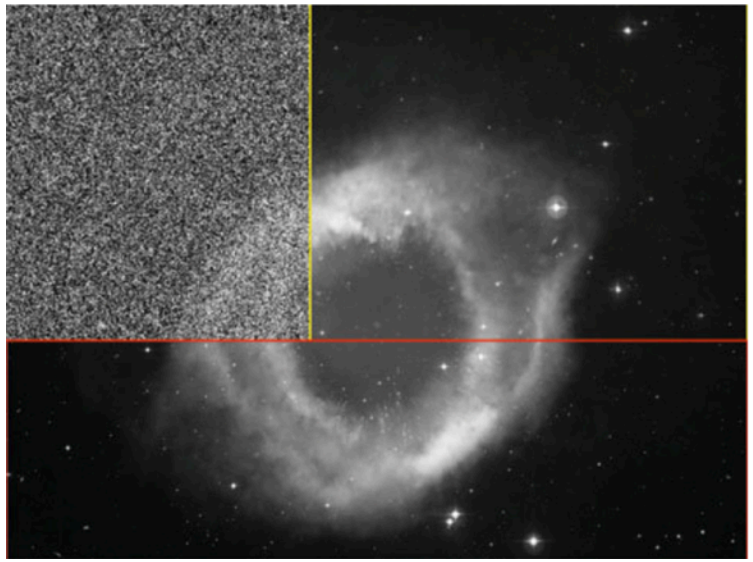


Can we use Pytorch to implement the minimization ?

No need to derive analytically the gradient of the variational cost ?

Inverse problems stated as minimisation problems

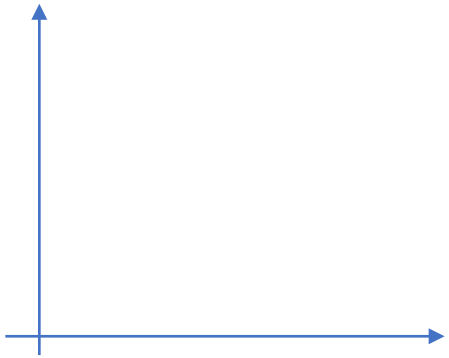
Denoising problem



Minimisation problem

$$X = \arg\min_X \|Y - H(X)\|^2 + \lambda U_{reg}(X)$$

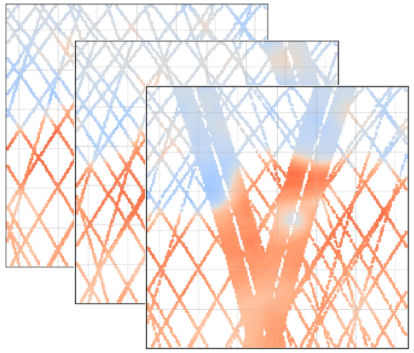
How to solve the minimisation ?



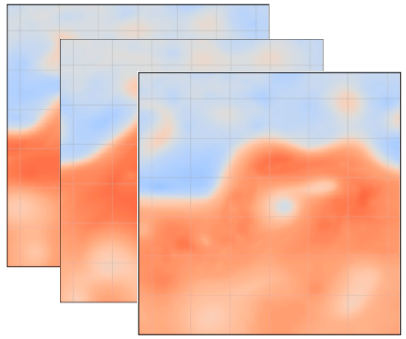
Can we use Pytorch to implement the minimization ?

No need to derive analytically the gradient of the variational cost ?

(Variational) Data Assimilation: (Weak-Constraint) 4DVar DA



Partial observations y



True states x

State-space formulation:

$$\begin{cases} \dfrac{\partial x(t)}{\partial t} &= \mathcal{M}\left(x(t)\right) + \eta(t) \\ \\ y(t,p) &= x(t,p) \quad \forall t, \forall p \in \Omega_t \end{cases}$$

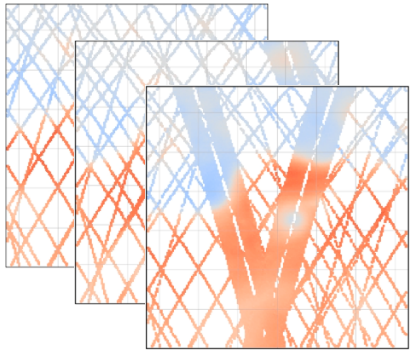
Associated variational formulation:

$$\arg\min_x \lambda_1 \sum_i \|x(t_i) - y(t_i)\|^2_{\Omega_{t_i}} + \lambda_2 \sum_n \|x(t_i) - \Phi(x)(t_i)\|^2$$

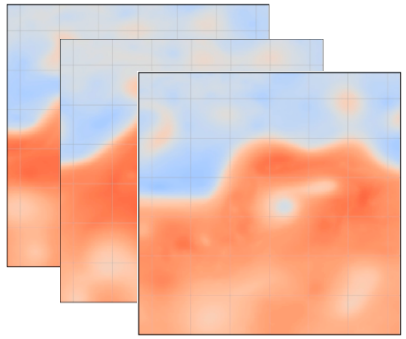
with $$\Phi(x)(t) = x(t-\Delta) + \int_{t-\Delta}^{t} \mathcal{M}\left(x(u)\right) du$$

$$\arg\min_x \lambda_1 \|x - y\|^2_\Omega + \lambda_2 \|x - \Phi(x)\|^2$$

(Variational) Data Assimilation: (Weak-Constraint) 4DVar DA



Partial observations y



True states x

Minimization problem

$$\arg\min_x \lambda_1 \|x - y\|_\Omega^2 + \lambda_2 \|x - \Phi(x)\|^2$$

with $$\Phi(x)(t) = x(t - \Delta) + \int_{t-\Delta}^{t} \mathcal{M}\left(x(u)\right) du$$

If the dynamical model is known, which implementation using a differentiable framework ?

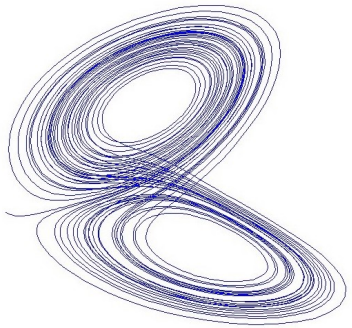
How to implement flow operator Φ ?

(Variational) Data Assimilation: (Weak-Constraint) 4DVar DA

Numerical integration scheme as residual neural networks (eg, Fablet et al., 2018)

$$\begin{aligned}\frac{\mathrm{d}x(t)}{\mathrm{d}t} &= \sigma\left(y(t)-x(t)\right)\\ \frac{\mathrm{d}y(t)}{\mathrm{d}t} &= x(t)\left(\rho-z(t)\right)-y(t)\\ \frac{\mathrm{d}z(t)}{\mathrm{d}t} &= x(t)\,y(t)-\beta\,z(t)\end{aligned}$$

Lorenz-63 equations

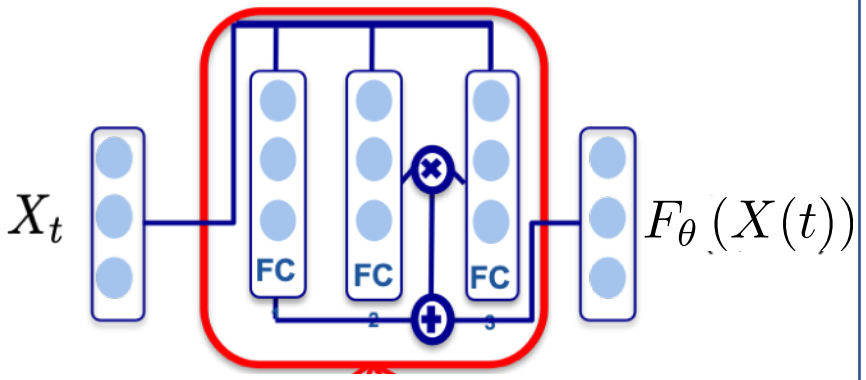


Associated Euler integration scheme

$$d_t X(t) = F_\theta\left(X(t)\right)$$

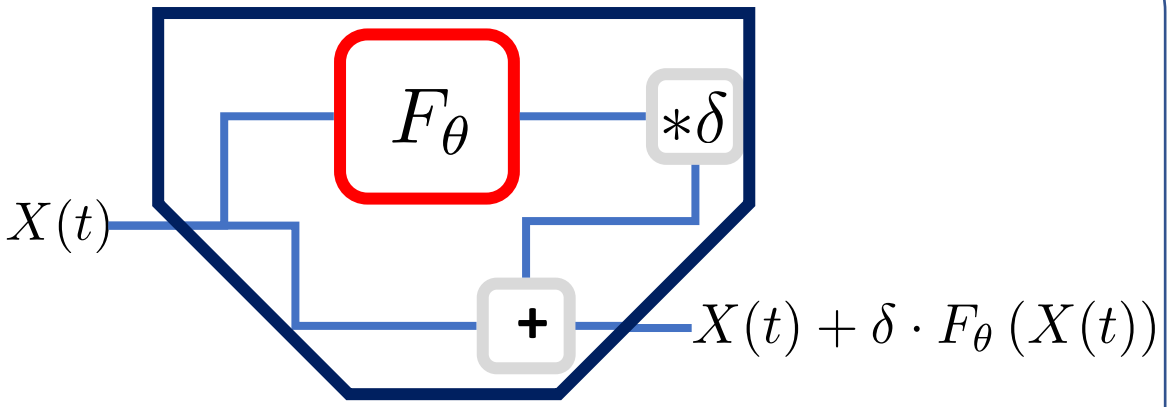
$$X(t+\delta) = X(t) + \delta \cdot F_\theta\left(X(t)\right)$$

NN architecture for differential operator



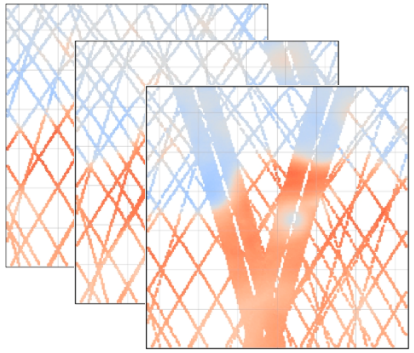
Bilinear architecture

NN architecture for integration scheme

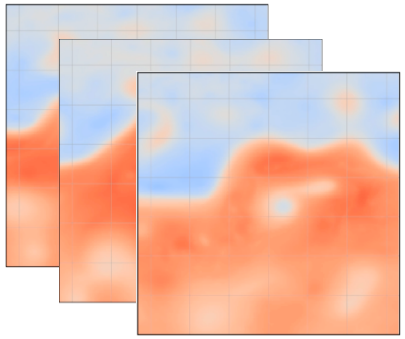


ResNet architecture (Residual Network)

(Variational) Data Assimilation: (Weak-Constraint) 4DVar DA



Partial observations y



True states x

Minimization problem

$$\arg\min_x \lambda_1 \|x - y\|_\Omega^2 + \lambda_2 \|x - \Phi(x)\|^2$$

with $$\Phi(x)(t) = x(t - \Delta) + \int_{t-\Delta}^{t} \mathcal{M}(x(u))\, du$$

Let's try to implement this minimisation with Pytorch.

https://github.com/CIA-Oceanix/DLGD2021/blob/main/lecture-5-inverse-problems/notebookPyTorch_InvProb_ModelBased_L63_Students.ipynb

Key message for (variational) Model-based Methods

- ***Solution stated as the minimisation of a variational cost***
- ***Explicit knowledge of observation model and prior***
- ***Implementation of associated minimisation algorithm***
- ***Analytical derivation of gradient operator (***e.g., Euler-Lagrange equations, Adjoint Methods***)***
- ***Implementation using DL schemes (***eg, Pytorch***) do not require the analytical gradient operator***

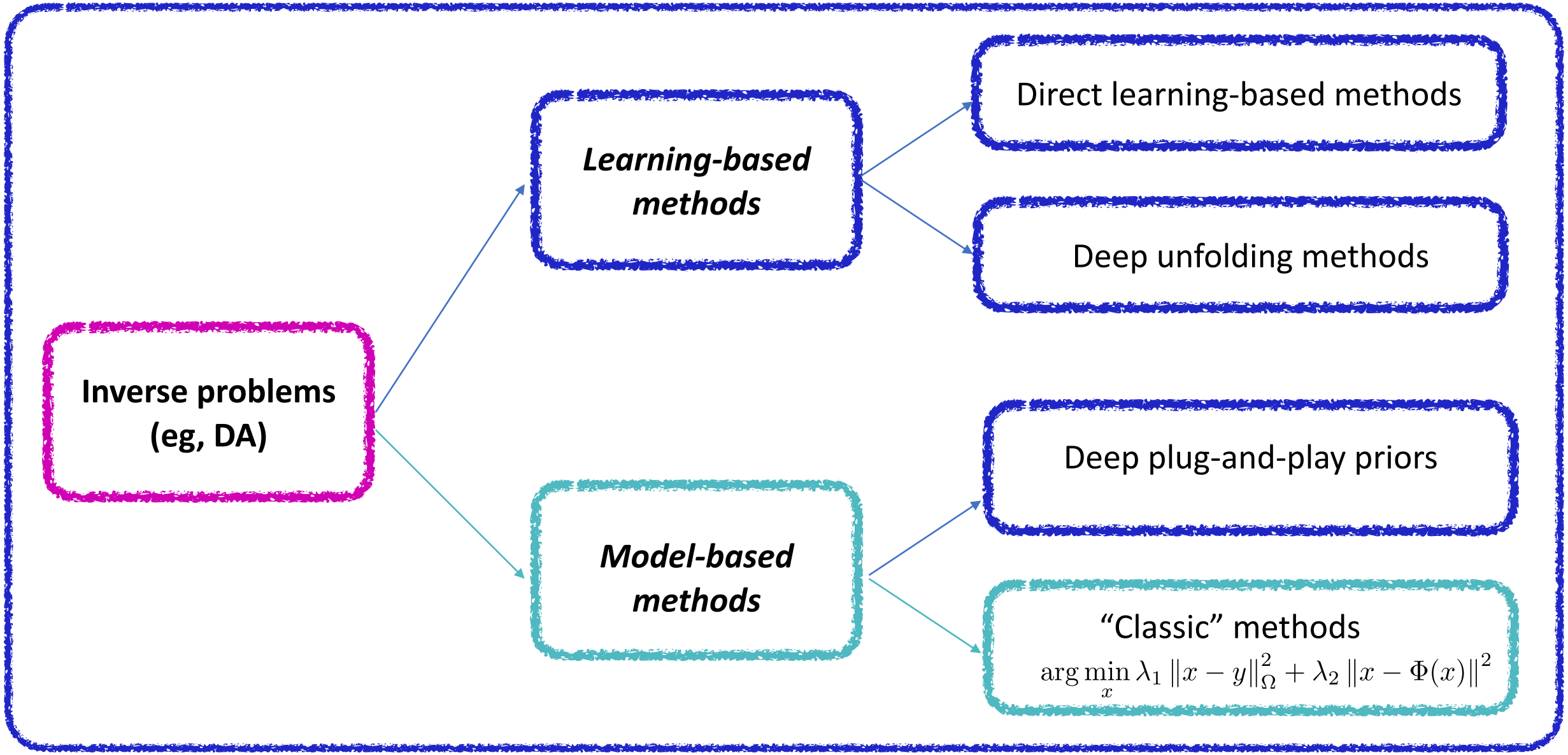
Inverse Problems in Geoscience

Mathematical formulations for inverse Problems

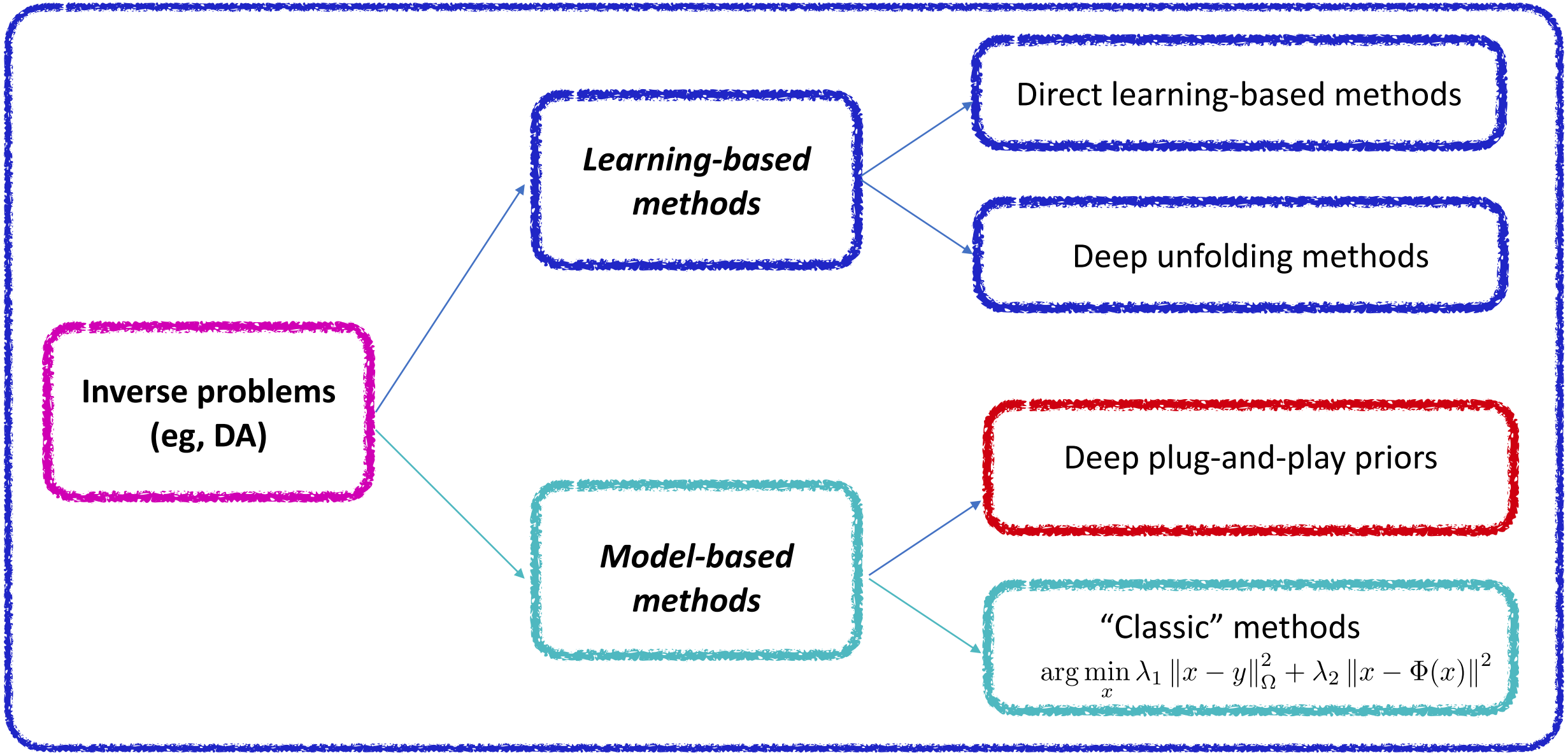
Inverse problems & Deep learning

Applications to geophysical dynamics

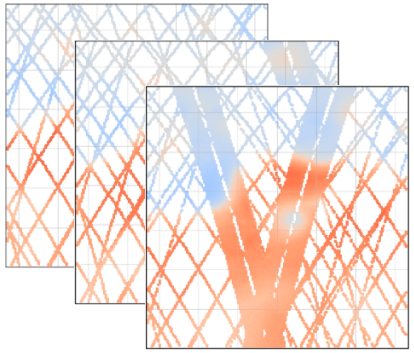
Model-driven vs. Learning-based approaches



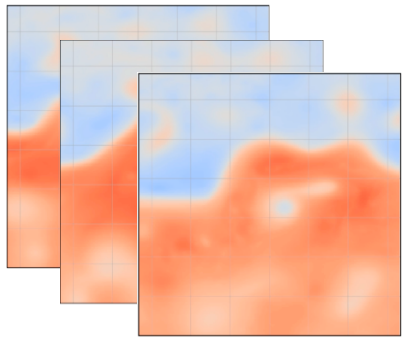
Model-driven vs. Learning-based approaches



Inverse problems using Deep plug-and-play priors



Partial observations y

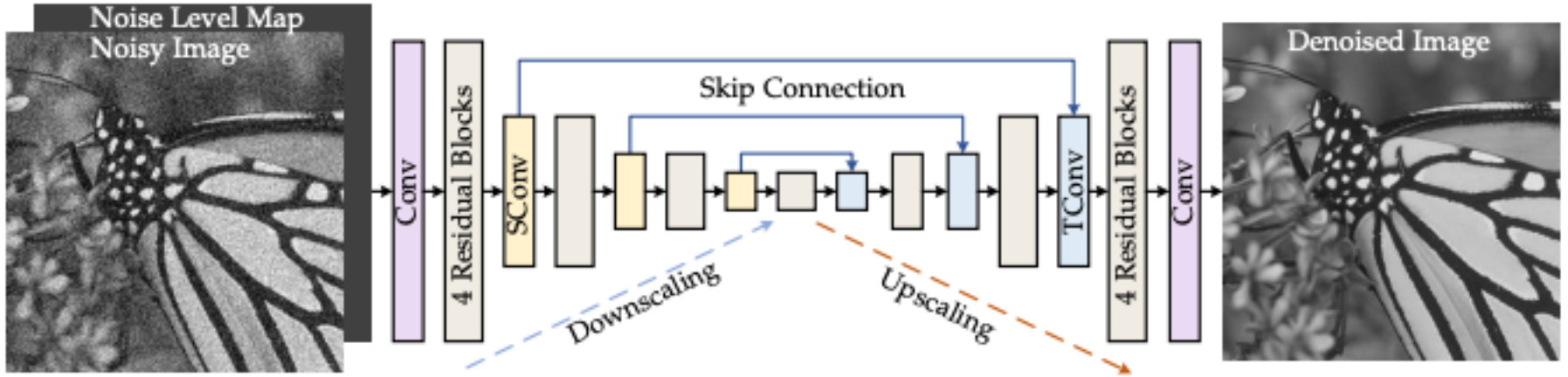


True states x

Model-based formulation

$$\arg\min_x \lambda_1 \|x - y\|_\Omega^2 + \lambda_2 \|x - \Phi(x)\|^2$$

Trainable plug-and-play prior



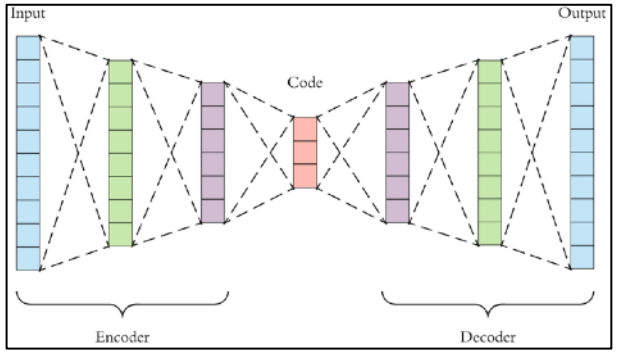
x

$\Phi(x)$

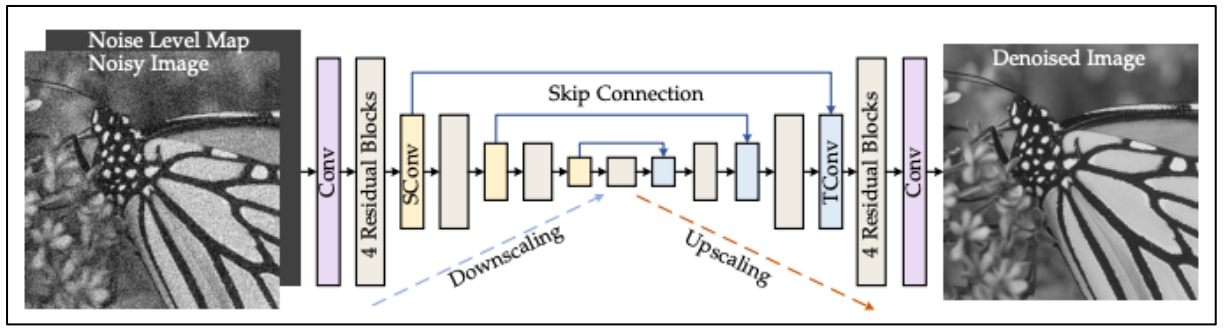
Zhang et al. Plug-and-Play Image Restoration with Deep Denoiser Prior. IEEE TPAMI, 2021. https://arxiv.org/pdf/2008.13751.pdf

Inverse problems using Deep plug-and-play priors

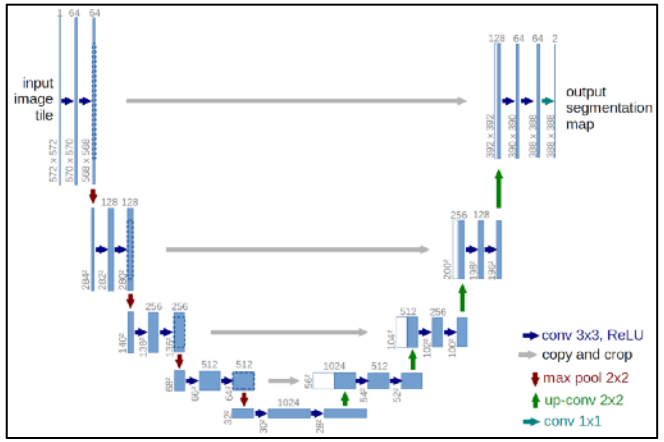
Examples of plug-and-play priors (denoiser architecture)



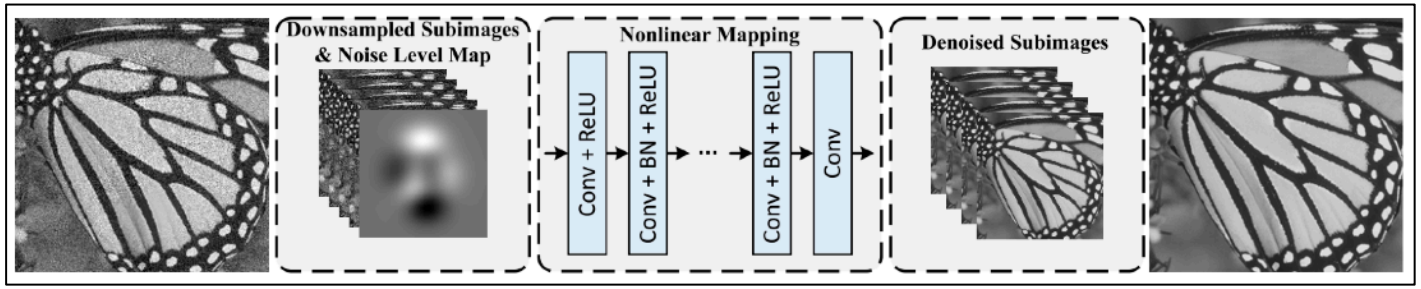
Auto-encoder



DRUNet https://arxiv.org/pdf/2008.13751.pdf

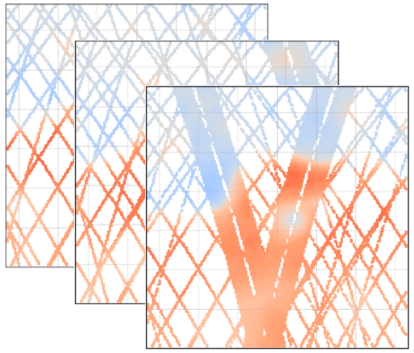


UNet https://arxiv.org/abs/1505.04597

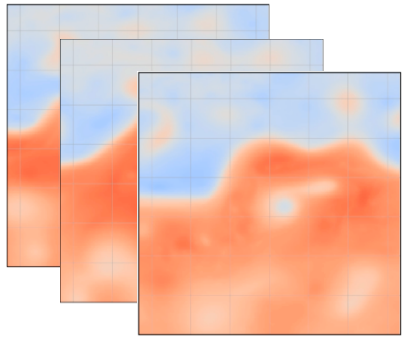


FFDNet https://arxiv.org/pdf/1710.04026.pdf

Inverse problems using Deep plug-and-play priors



Partial observations y



True states x

Model-based formulation

$$\arg\min_x \lambda_1 \|x - y\|_\Omega^2 + \lambda_2 \|x - \Phi(x)\|^2$$

Trainable plug-and-play prior

Use of trainable priors but no actual learning specifically designed for the targeted inverse problem

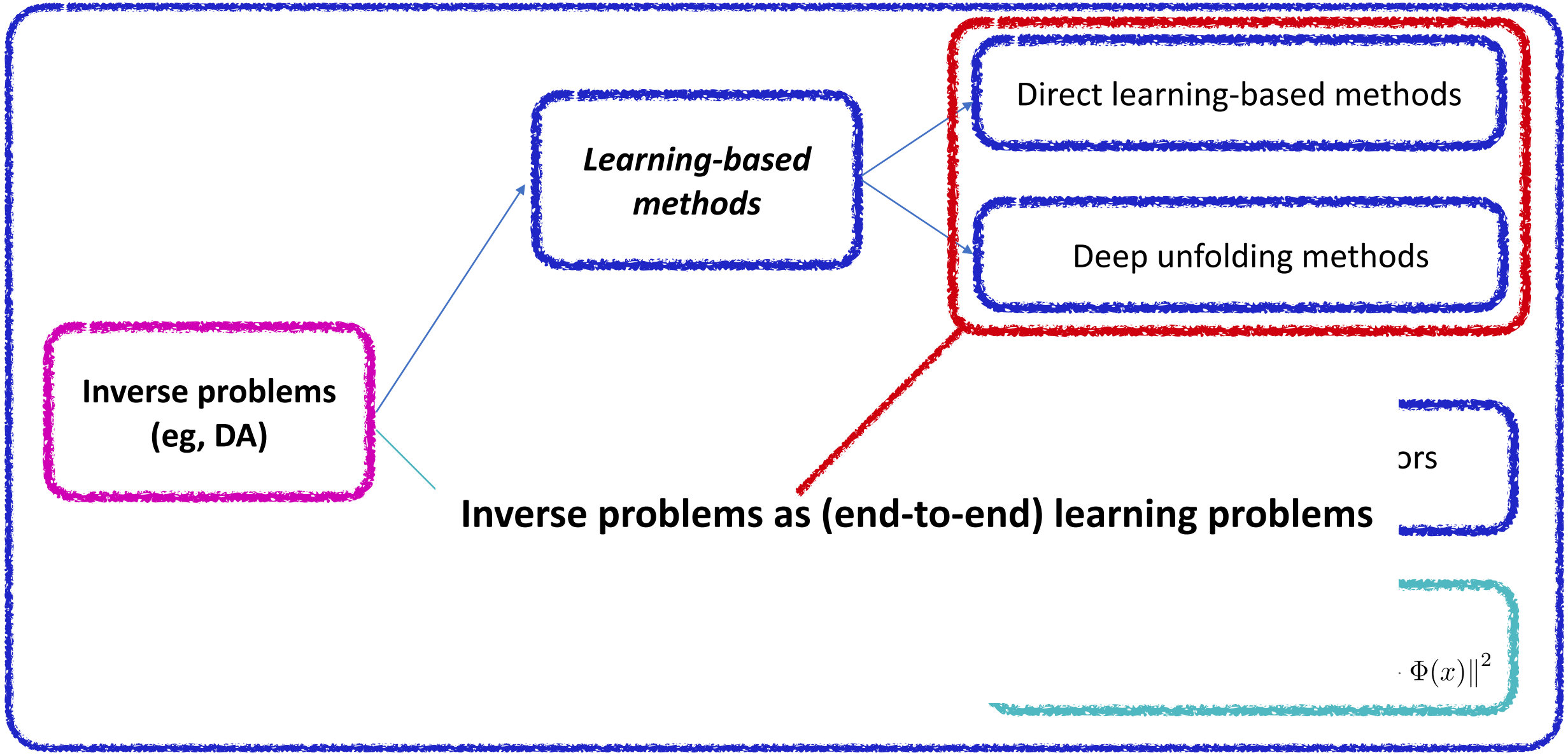
Let's go and test it using a PCA-based prior

https://github.com/CIA-Oceanix/DLGD2021/blob/main/lecture-5-inverse-problems/notebookPyTorch_InvProb_ModelBased_L63_Students.ipynb

Key message for (variational) Model-based Methods

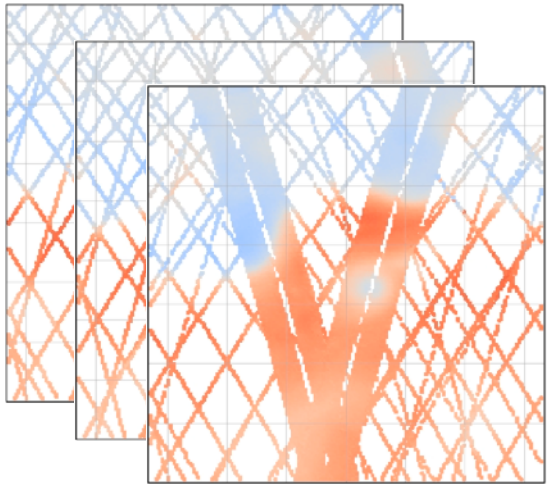
- ***Solution stated as the minimisation of a variational cost***
- ***Explicit knowledge of observation model and prior***
- ***Implementation of associated minimisation algorithm***
- ***Analytical derivation of gradient operator (***e.g., Euler-Lagrange equations, Adjoint Methods***)***
- ***Implementation using DL schemes (***eg, Pytorch***) do not require the analytical gradient operator***
- ***Possible extension to pre-trained plug-and-play priors***

Model-driven vs. Learning-based approaches



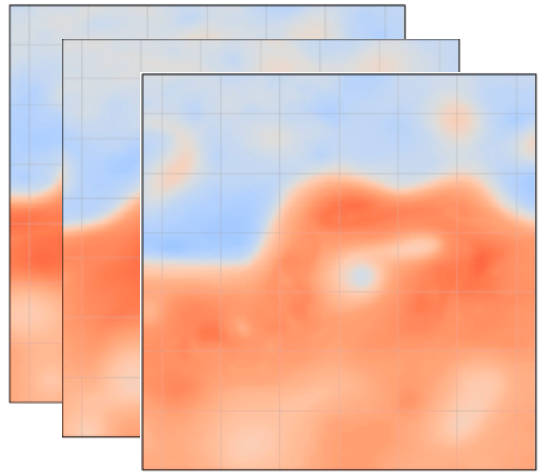
End-to-end learning for inverse problems

End-to-end architecture



Partial observations y

Deep learning Models

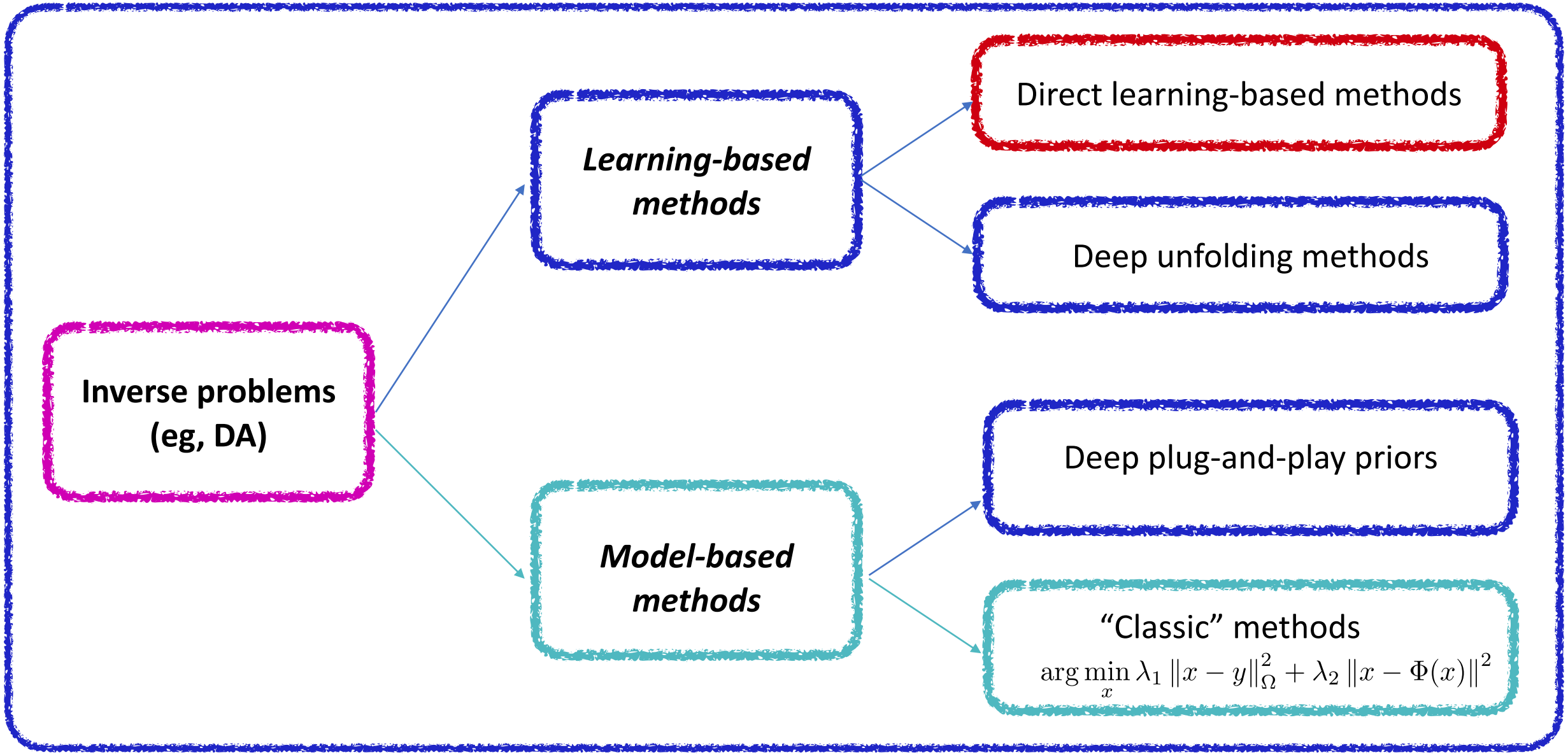


True states x

Which training loss ?

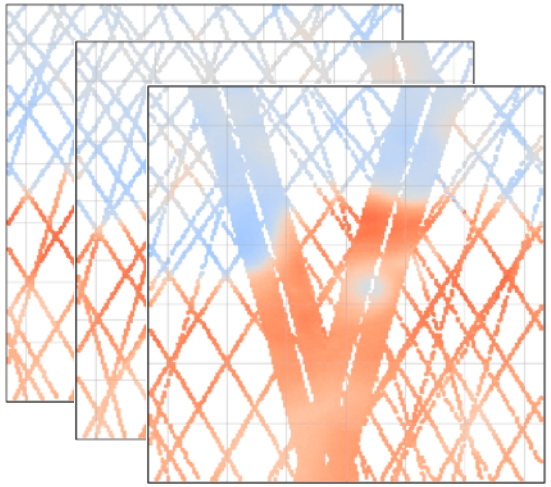
Which models / architectures ?

Model-driven vs. Learning-based approaches

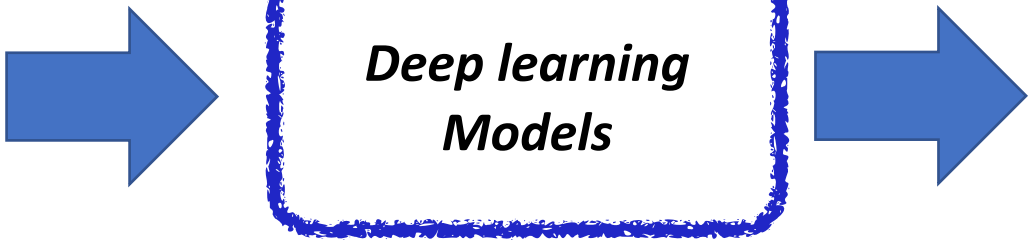


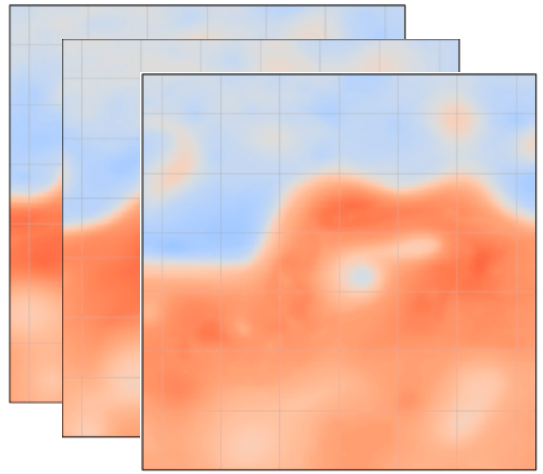
End-to-end learning for inverse problems

End-to-end architecture



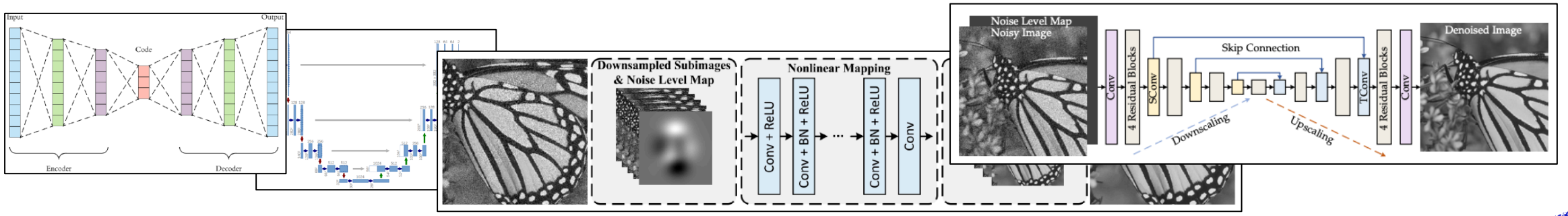
Partial observations y





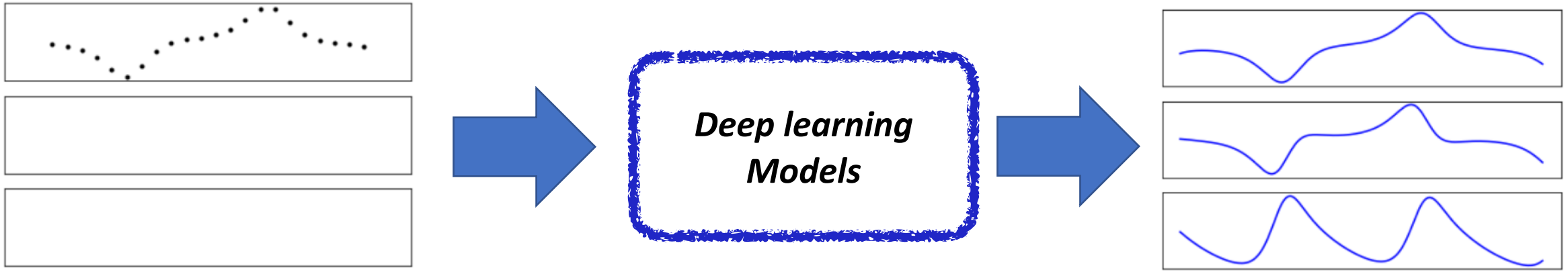
True states x

Which architectures? State-of-the-art CNN architectures?



End-to-end learning for inverse problems

An illustration for Lorenz-63 dynamics



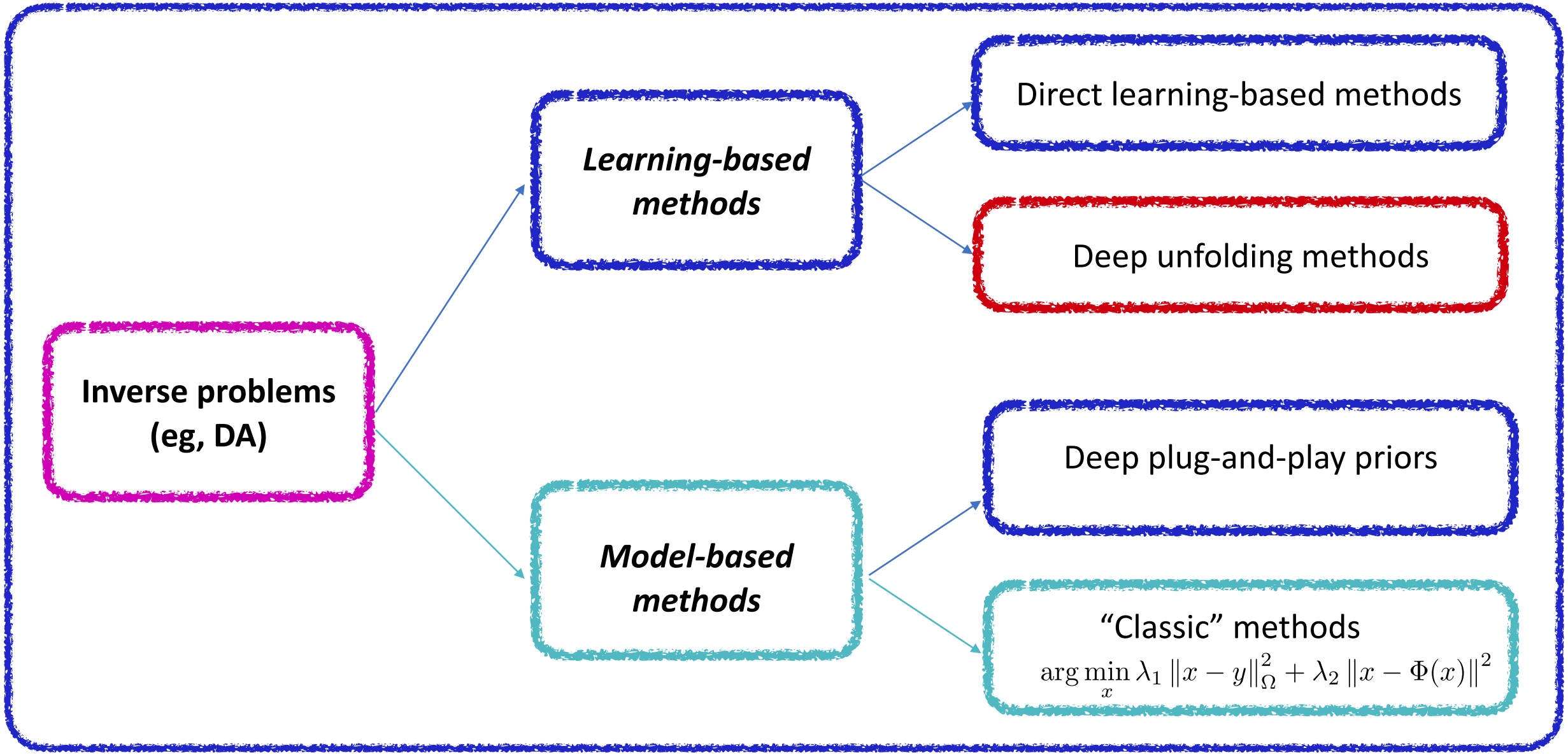
Partial observations y

True states x

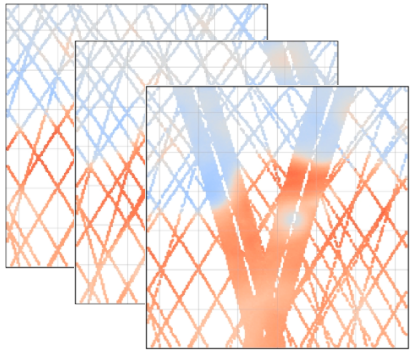
Colab notebook:

https://colab.research.google.com/drive/1mQVBUS-zQUIdXieJFkrpaC6yNkoMvKpR?usp=sharing

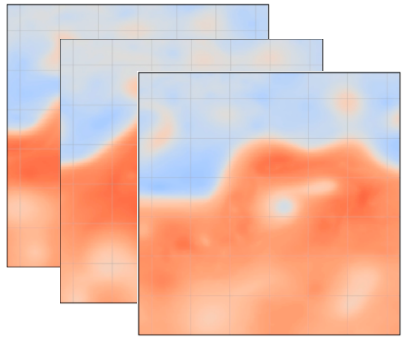
Model-driven vs. Learning-based approaches



Inverse problems using Deep unfolding schemes



Partial observations y



True states x

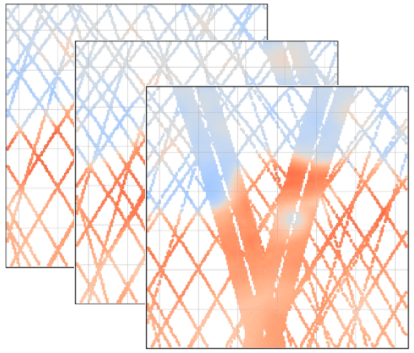
Basic idea:

- **Many schemes involve iterative algorithms**
- **On may unfold an Iterative procedure to define a deep learning architecture**

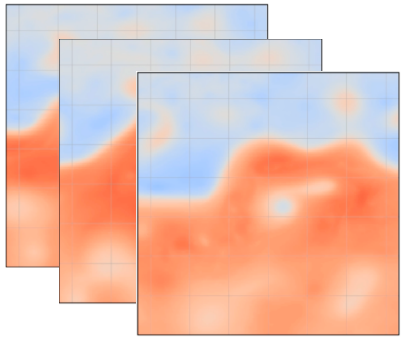
Examples

- **Unfolding of reaction-diffusion schemes** (e.g., Chen et al., 2015)
- **Unfolding of ADMM schemes** (e.g., Yang et al., 2016)
- **Unfolding of fixed-point algorithms** (e.g., Fablet et al., 2020)

Inverse problems using Deep unfolding schemes



Partial observations y



True states x

An example using fixed-point algorithms for interpolation problems

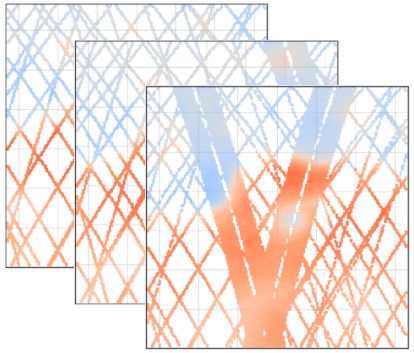
$$\arg\min_x \|x - \Phi(x)\|^2 \text{ subject to } y_\Omega = x_\Omega$$

Iterative step

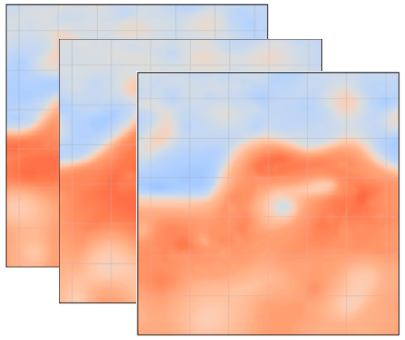
$$\begin{cases} \tilde{x}^{(k)} = \Phi\left(x^{(k)}\right) & \text{Projection using } \Phi \\ x^{(k+1)}(\Omega) = y^{(k)}(\Omega) & \text{Keep observed data} \\ x^{(k+1)}\left(\overline{\Omega}\right) = \tilde{x}^{(k)}\left(\overline{\Omega}\right) & \text{Update unobserved data} \end{cases}$$

Associated Deep unfolding scheme

Inverse problems using Deep unfolding schemes



Partial observations y



True states x

An example using fixed-point algorithms for interpolation problems

$$\arg\min_x \|x - \Phi(x)\|^2 \text{ subject to } y_\Omega = x_\Omega$$

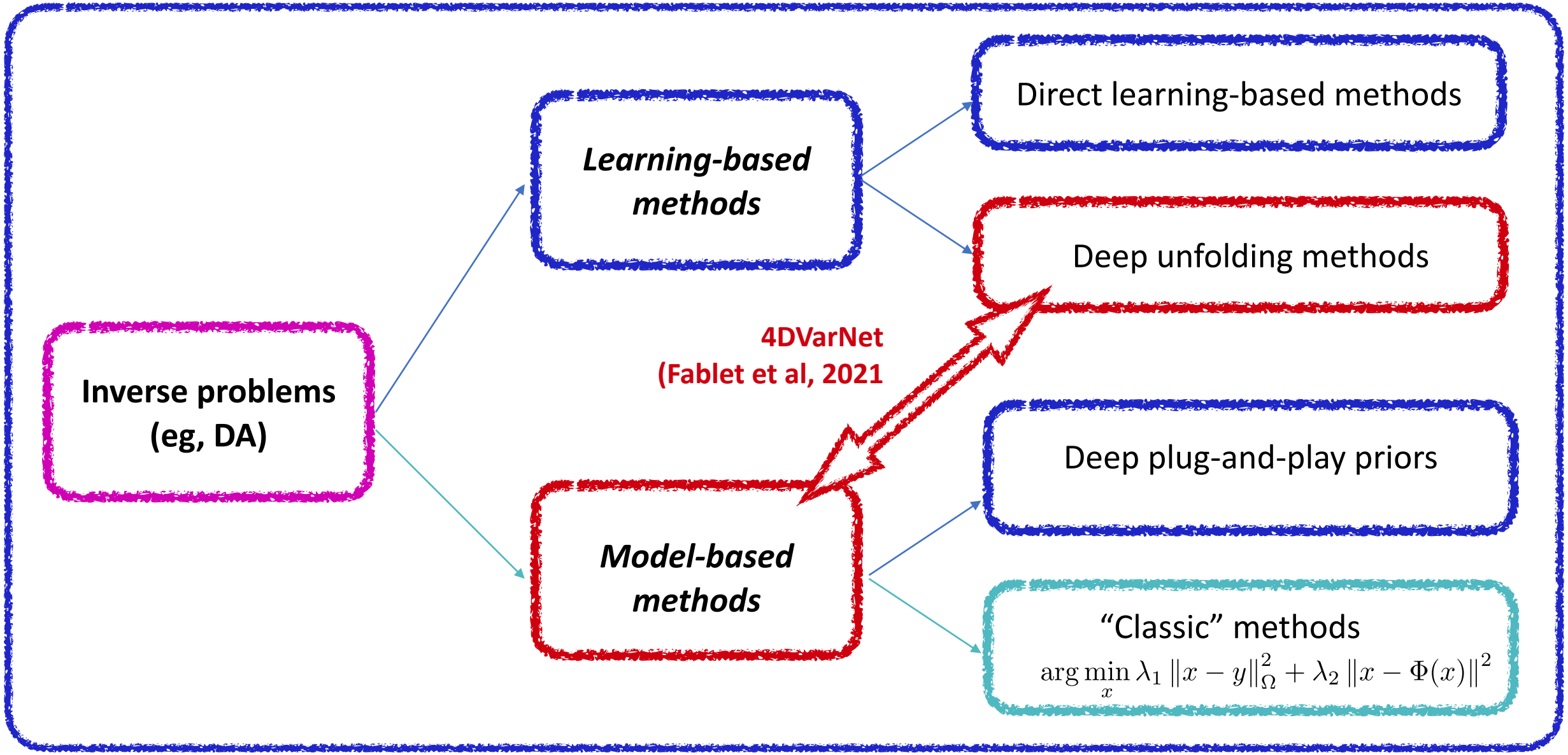
Iterative step

$$\begin{cases} \tilde{x}^{(k)} = \Phi\left(x^{(k)}\right) & \text{Projection using } \Phi \\ x^{(k+1)}(\Omega) = y^{(k)}(\Omega) & \text{Keep observed data} \\ x^{(k+1)}(\overline{\Omega}) = \tilde{x}^{(k)}(\overline{\Omega}) & \text{Update unobserved data} \end{cases}$$

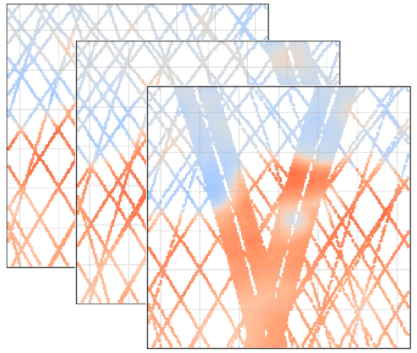
Associated Deep unfolding scheme

https://colab.research.google.com/drive/1mQVBUS-zQUIdXieJFkrpaC6yNkoMvKpR?usp=sharing

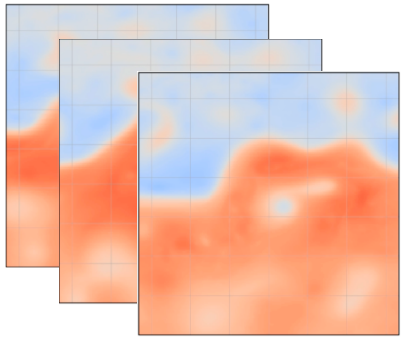
Model-driven vs. Learning-based approaches



Inverse problems: Unfolding 4DVar DA



Partial observations y



True states x

Gradient-based solver (adjoint/Euler-Lagrange method):

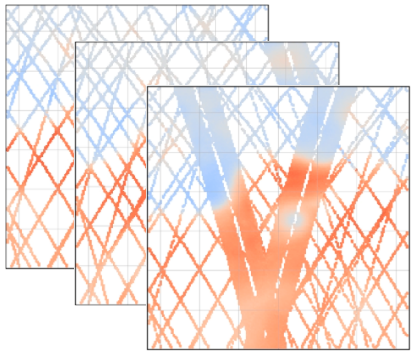
$$\arg\min_x \underbrace{\lambda_1 \|x-y\|_\Omega^2 + \lambda_2 \|x-\Phi(x)\|^2}_{U_\Phi(x,y,\Omega)}$$

Iterative update:

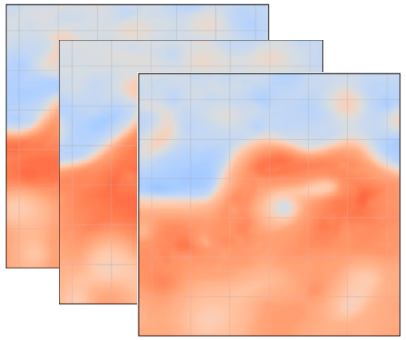
$$x^{(k+1)} = x^{(k)} - \alpha \nabla_x U_\Phi \left(x^{(k)}, y, \Omega \right)$$

Associated residual architecture ?

4DVar Data Assimilation (DA) formulation



Partial observations y



True states x

Model-driven schemes:

$$\arg\min_x \underbrace{\lambda_1 \|x-y\|_\Omega^2 + \lambda_2 \|x-\Phi(x)\|^2}_{U_\Phi(x,y,\Omega)}$$

Gradient-based solver (adjoint/Euler-Lagrange method):

$$x^{(k+1)} = x^{(k)} - \alpha \nabla_x U_\Phi\left(x^{(k)}, y, \Omega\right)$$

Implementation of 4DVar Data Assimilation using Deep Learning frameworks

- Implement operator Φ as a neural network (e.g., neural ODE architecture w.r.t. a known dynamical operator)
- Use embedded automatic differentiation tool to implement the above gradient descent without deriving analytically the adjoint operator

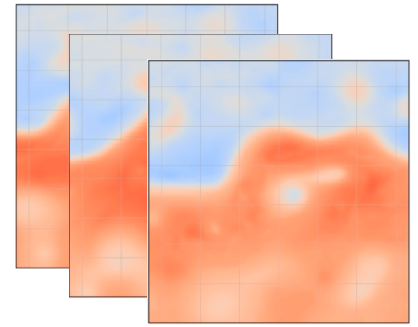
Inverse problems: Unfolding 4DVar DA

Iterative update: $x^{(k+1)} = x^{(k)} - \alpha \nabla_x U_\Phi \left(x^{(k)}, y, \Omega \right)$

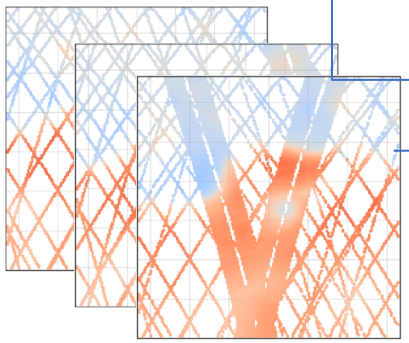
Initial state $x^{(0)}$

ResNet architecture

RU $\quad x^{(1)}$ $(....)$ RU $\quad x^{(K)}$ RU $\quad \tilde{x}$



Reconstructed states x



Observation data y, Ω

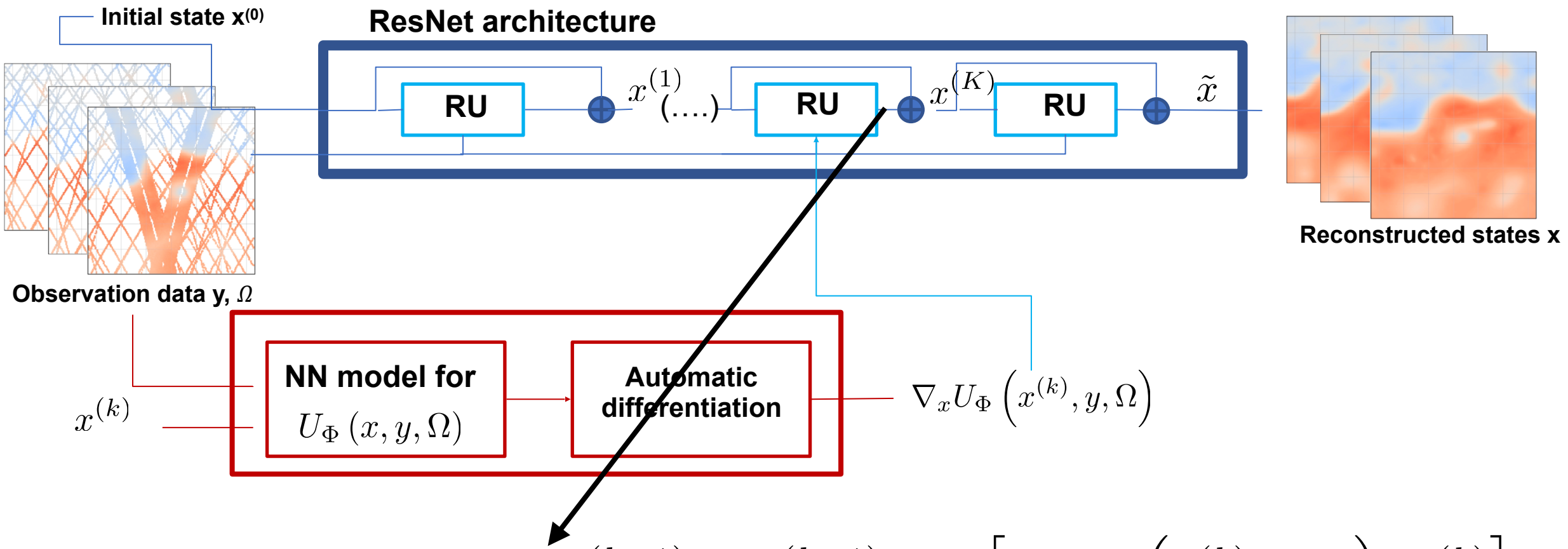
$x^{(k)}$

NN model for $U_\Phi (x, y, \Omega)$

Automatic differentiation

$\nabla_x U_\Phi \left(x^{(k)}, y, \Omega \right)$

Inverse problems: Unfolding 4DVar DA and Meta-learning



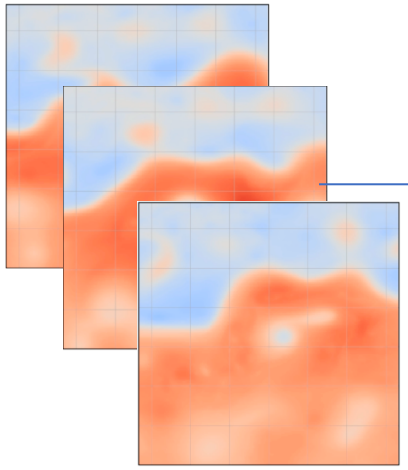
Generalised trainable update: $x^{(k+1)} = x^{(k+1)} + \mathcal{L}\left[\nabla_x U_\Phi\left(x^{(k)}, y, \Omega\right), z^{(k)}\right]$

Similar to learning-to-learn idea (*Andrychowicz et al., 2016*)

End-to-end learning for 4DVar DA: projection operator Φ

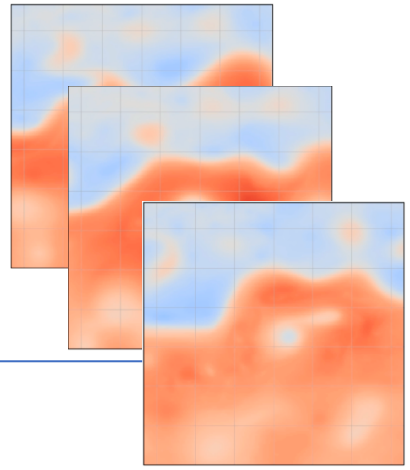
Parameterization using (learnable) ODE operator

$$\frac{\partial x(t)}{\partial t} = \mathcal{M}_\theta\left(x(t)\right)$$



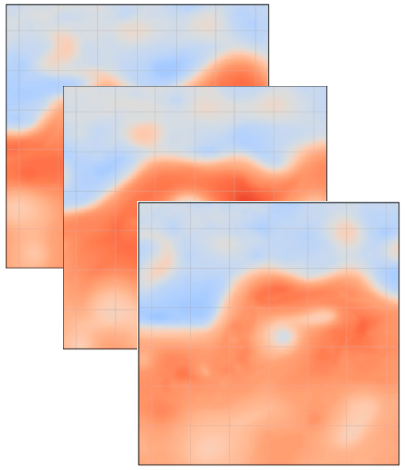
Neural integrator

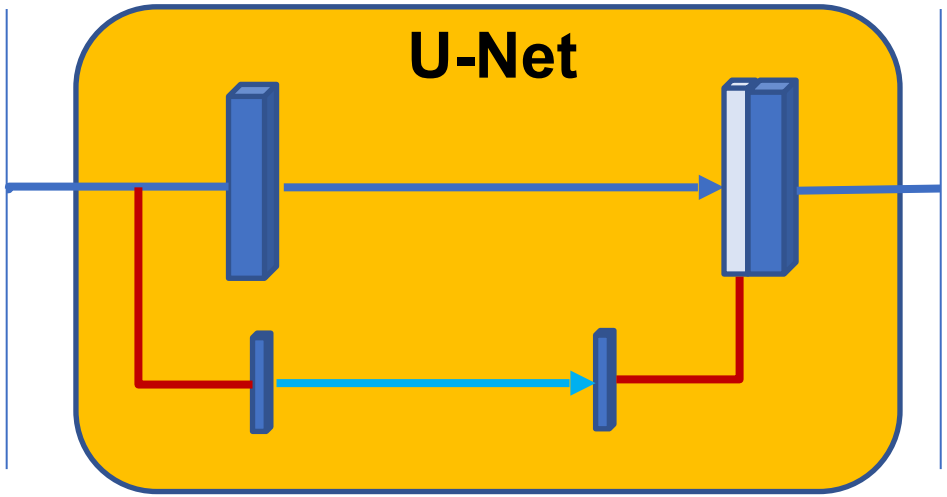
$$x(t-\Delta) + \int_{t-\Delta}^{t} \mathcal{M}_\theta\left(x(u)\right) du$$

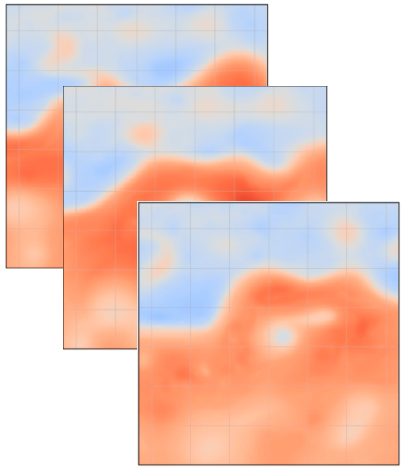


x $\Phi(x)$

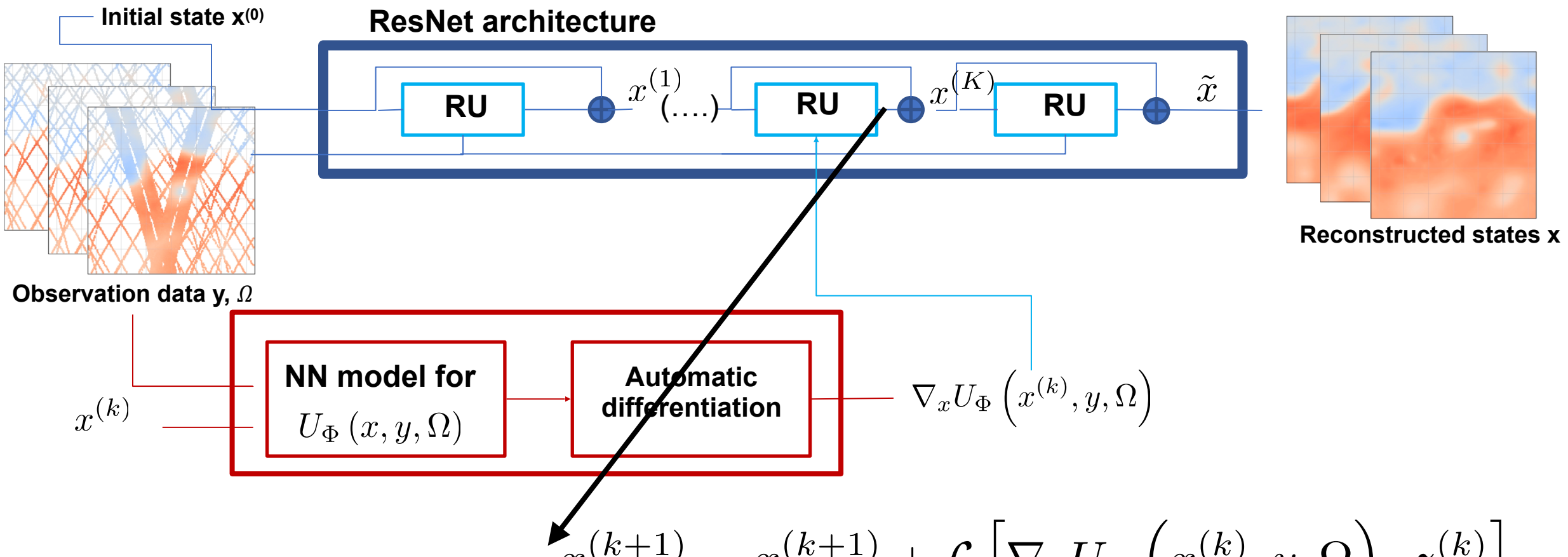
Two-scale U-Net-like Parameterization (Gibbs Field)







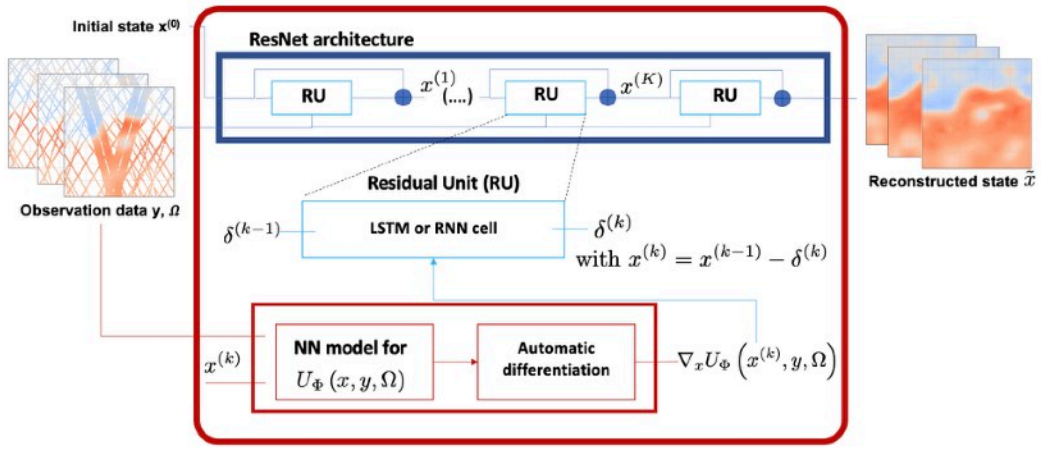
Inverse problems: Unfolding 4DVar DA and Meta-learning



Generalised trainable update: $x^{(k+1)} = x^{(k+1)} + \mathcal{L}\left[\nabla_x U_\Phi\left(x^{(k)}, y, \Omega\right), z^{(k)}\right]$

Colab notebook: https://colab.research.google.com/drive/1p0YA348fEh5Zy40dEs-AM-vukuJ_t_n2?usp=sharing

Model-based vs. Learning-based 4DVar DA: Unsupervised vs. Supervised scheme



Which training loss for 4DVarNet scheme ?

Unsupervised loss

$$\mathcal{L}(x, y) = \|x - y\|^2 + \lambda \|x - \Phi(x)\|^2$$

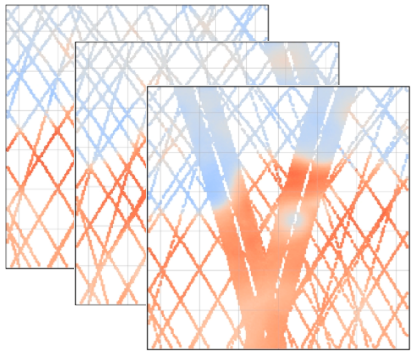
Supervised loss

$$\mathcal{L}\left(x, x^{true}\right) = \left\|x - x^{true}\right\|^2$$

Regularisation loss

$$\mathcal{L}_{Reg}\left(x, x^{true}\right) = \|x - \Phi(x)\|^2 + \left\|x^{true} - \Phi(x^{true})\right\|^2$$

Model-based vs. Learning-based 4DVar DA



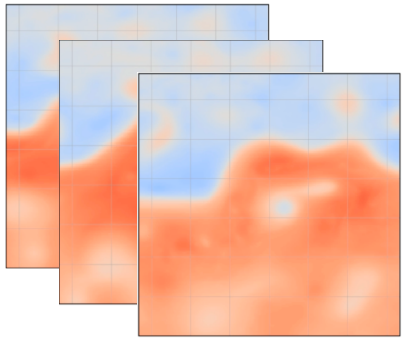
Partial observations y

Model-driven schemes: $\widehat{x} = \arg\min_x \underbrace{\lambda_1 \|x - y\|_\Omega^2 + \lambda_2 \|x - \Phi(x)\|^2}_{U_\Phi\left(x^{(k)}, y, \Omega\right)}$

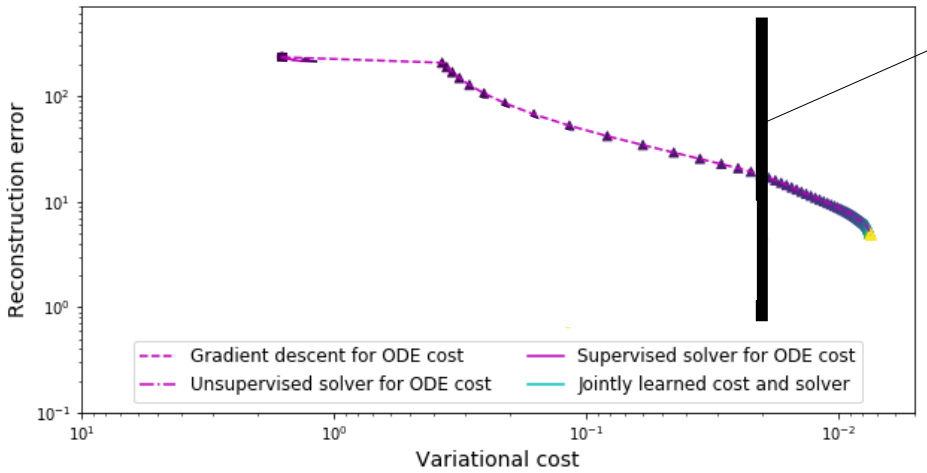
Gradient-based solver (adjoint/Euler-Lagrange method):

$$x^{(k+1)} = x^{(k)} - \alpha \nabla_x U_\Phi\left(x^{(k)}, y, \Omega\right)$$

No control on the reconstruction error $x^{true} \neq \arg\min_x U_\Phi\left(x^{(k)}, y, \Omega\right)$

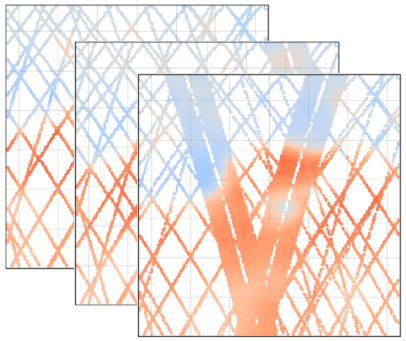


True states x

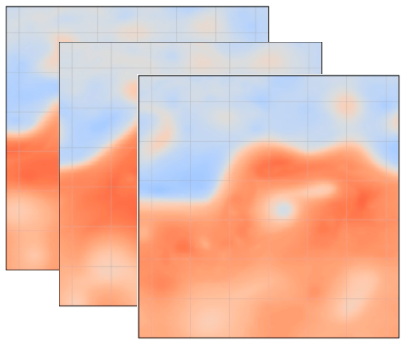


Variational cost for the true state

Model-based vs. Learning-based 4DVar DA



Partial observations y



True states x

Model-driven schemes: $\widehat{x} = \arg\min_x \lambda_1 \|x - y\|_\Omega^2 + \lambda_2 \|x - \Phi(x)\|^2$

Proposed scheme: joint learning of the variational model and solver

- **Theoretical bi-level optimization**

$$\arg\min_\Phi \sum_n \|x_n - \tilde{x}_n\|^2 \text{ s.t. } \tilde{x}_n = \arg\min_{x_n} U_\Phi(x_n, y_n, \Omega_n)$$

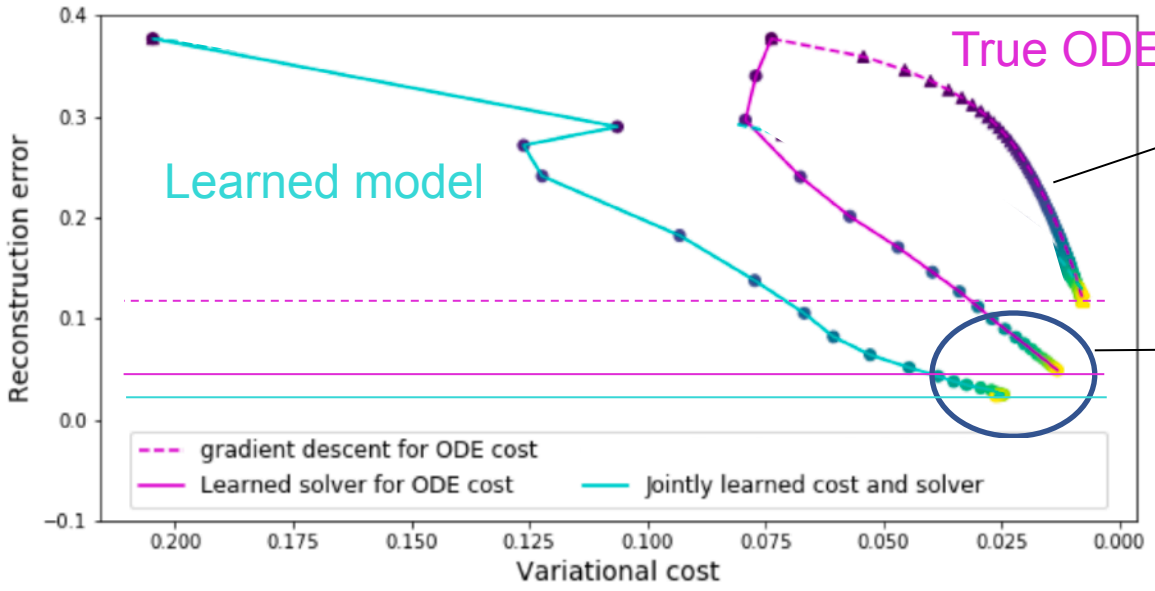
- **Restated with a gradient-based NN solver for inner minimization**

$$\arg\min_{\Phi,\Gamma} \sum_n \|x_n - \tilde{x}_n\|^2 \text{ s.t. } \tilde{x}_n = \Psi_{\Phi,\Gamma}(x_n^{(0)}, y_n, \Omega_n)$$

Iterative 4DVarNet solver using automatic differentiation to compute gradient $\nabla_x U_\Phi\left(x^{(k)}, y, \Omega\right)$

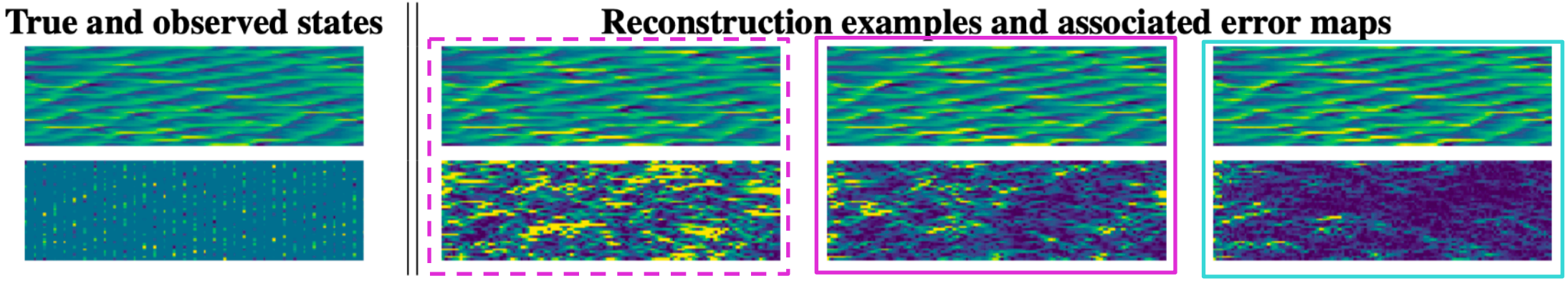
End-to-end learning for inverse problems (Fablet etal., 2020)

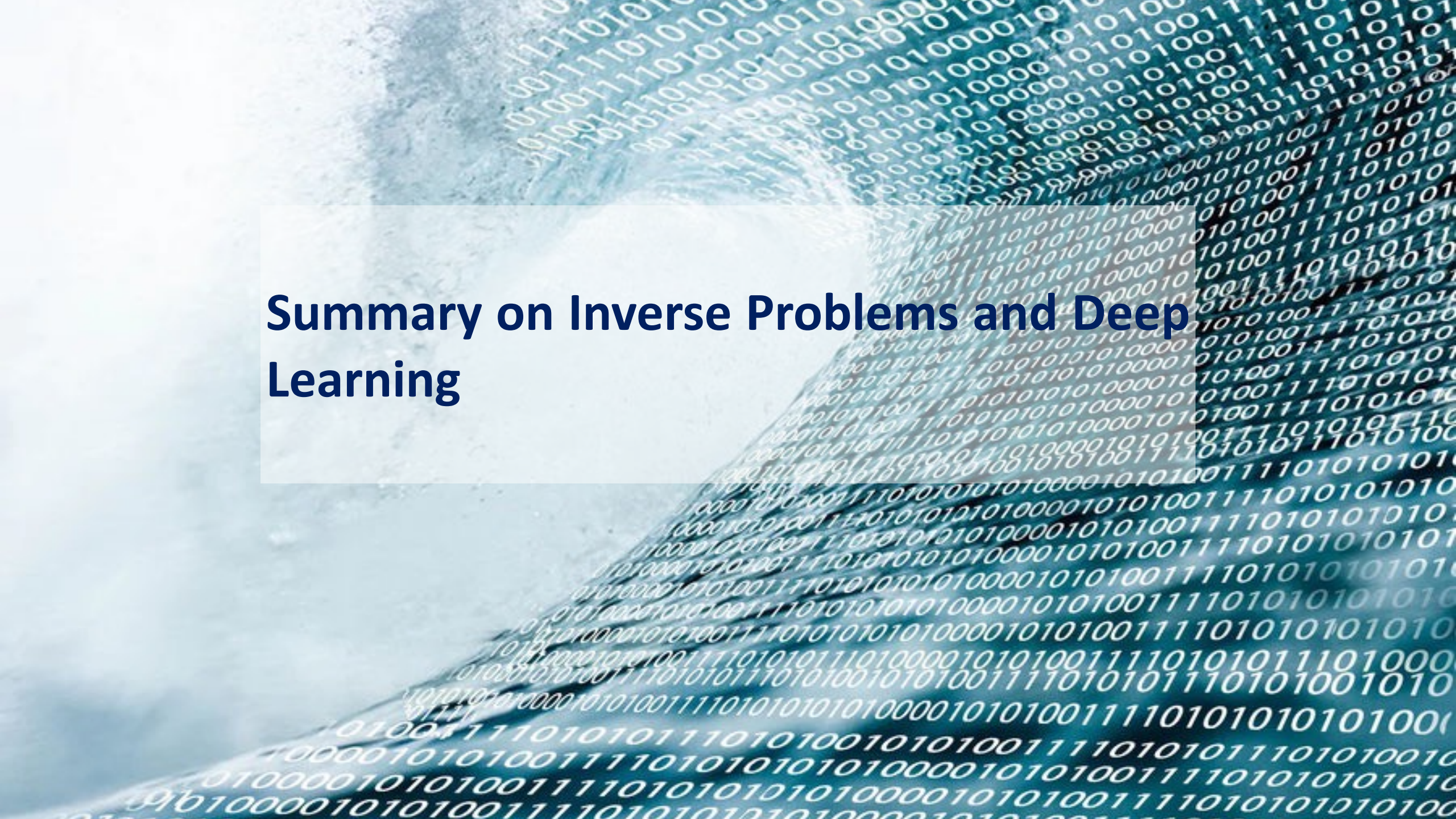
Illustration on Lorenz-96 dynamics (Bilinear ODE)



Non-supervised setting
(classic variational minimisation)

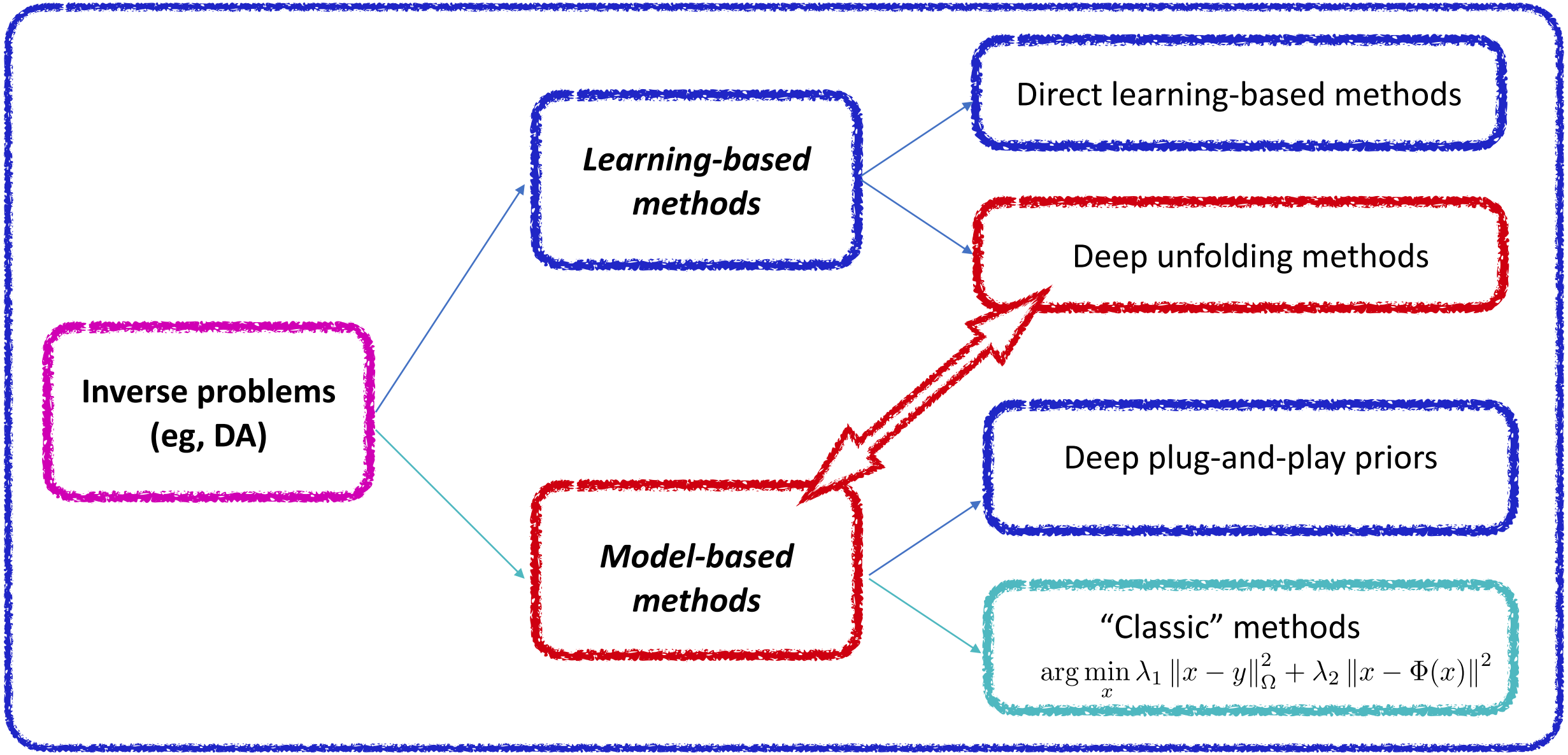
Supervised settings





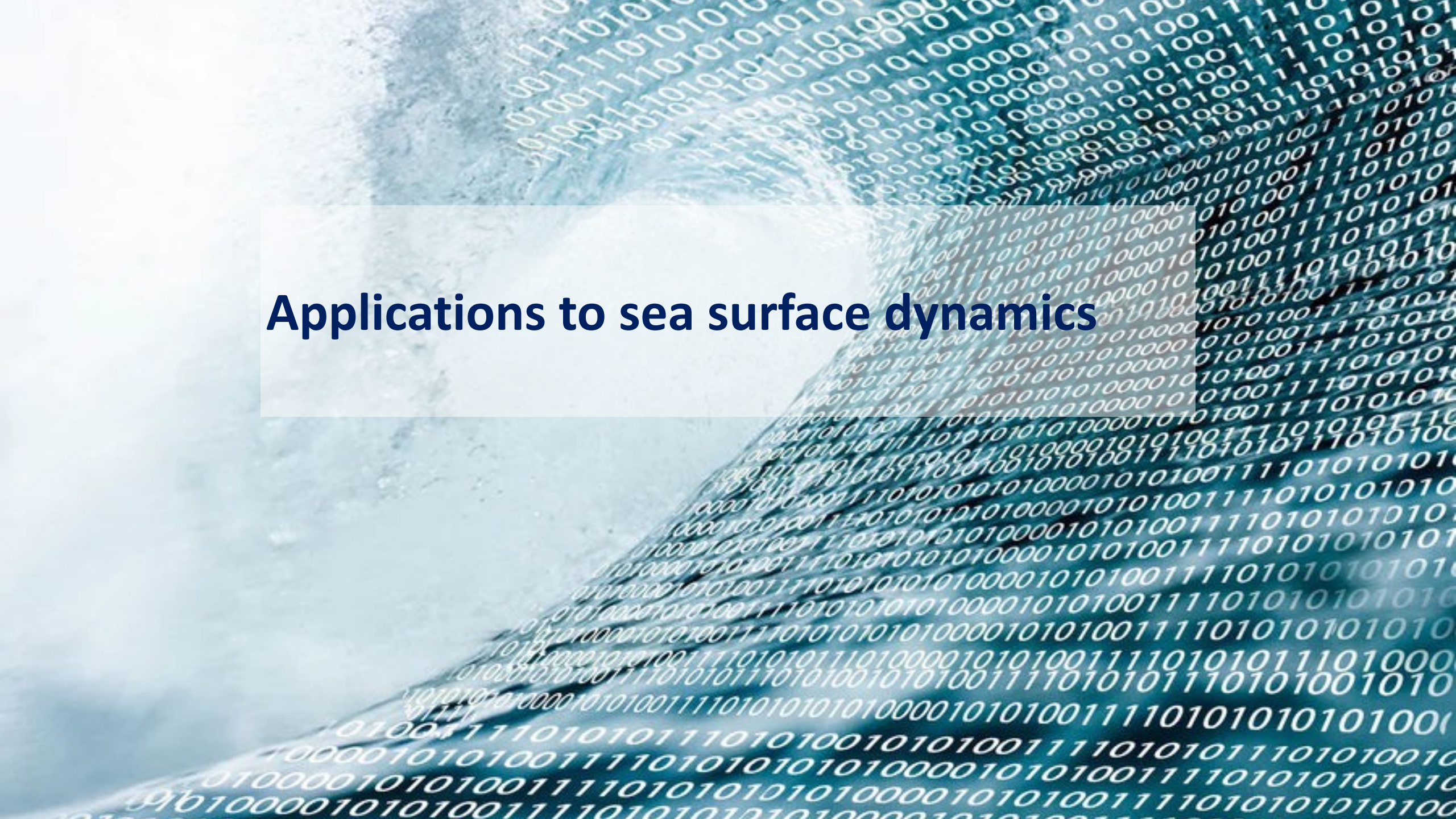
Summary on Inverse Problems and Deep Learning

Model-driven vs. Learning-based approaches



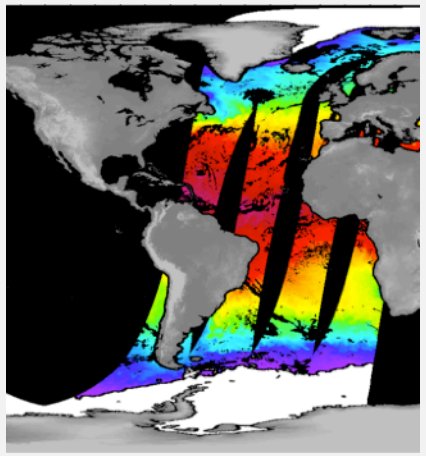
Summary

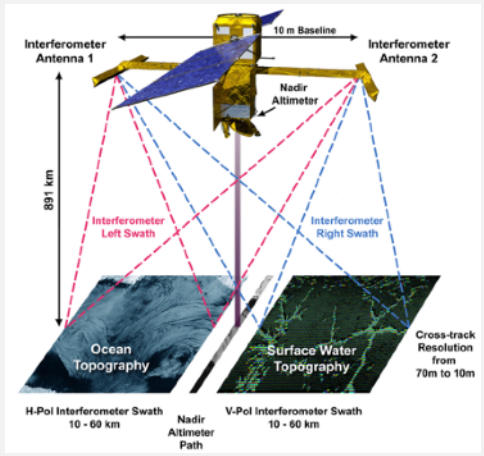
- ***NNs as numerical schemes for ODE/PDE/variational representations of geophysical dynamics***
- ***NN plug-and-play priors***
- ***End-to-end architecture for jointly learning a representation (***eg, ODE***) and a solver***
- ***Requirement for differential implementations***
- ***The true prior might not be the optimal choice to solve inverse problems***

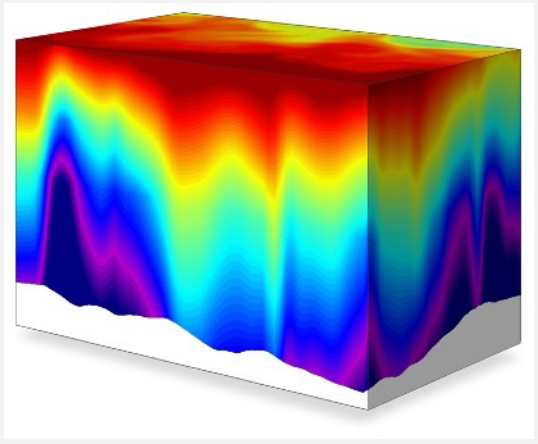


Applications to sea surface dynamics

Dealing with partially-observed and irregularly-sampled ocean dynamics







Problem statement:
Joint identification and inversion

Dynamical model

$$X_t \rightarrow \partial_t X = F(X, \xi, t, \theta) \rightarrow X_{t+1}$$

$+$

Observation model

$$Y_t = H(X, \zeta, t, \phi)$$

Fablet et al. **4DVarNet: Trainable Data Assimilation for Sea Surface dynamics**
https://cia-oceanix.github.io/

Method

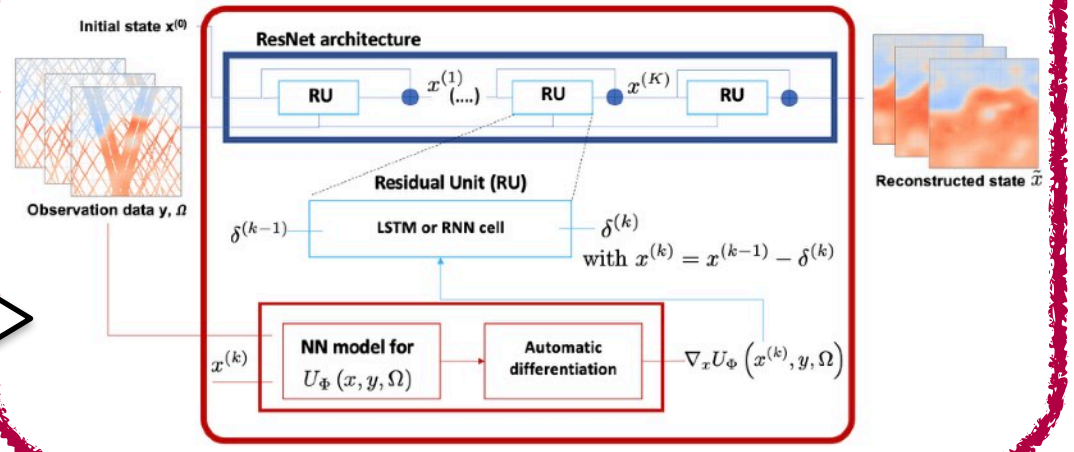
From a Variational DA formulation

$$\hat{x} = \arg\min_x \|x - y\|^2 + \lambda \|x - \phi(x)\|^2$$

Trainable variational model

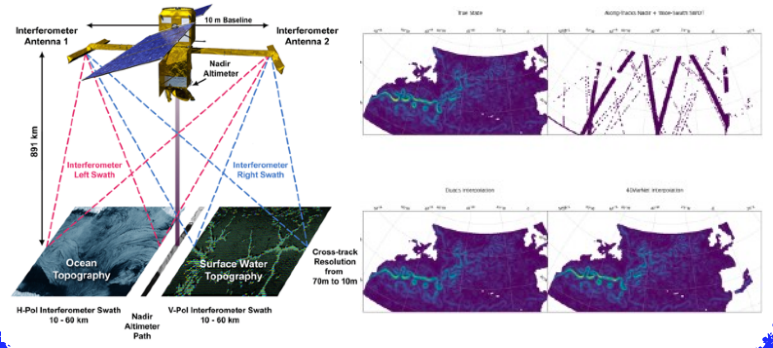
Trainable gradient-based solver

Associated end-to-end scheme

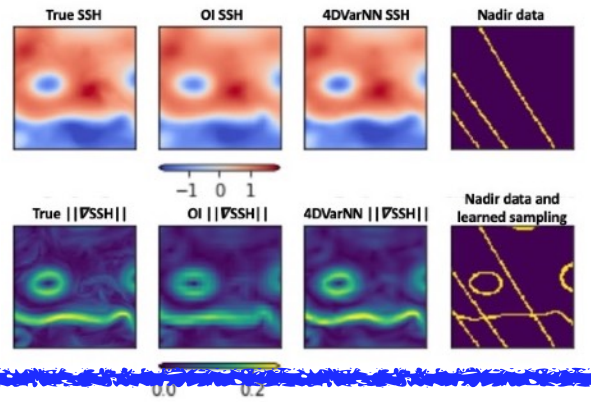


Applications

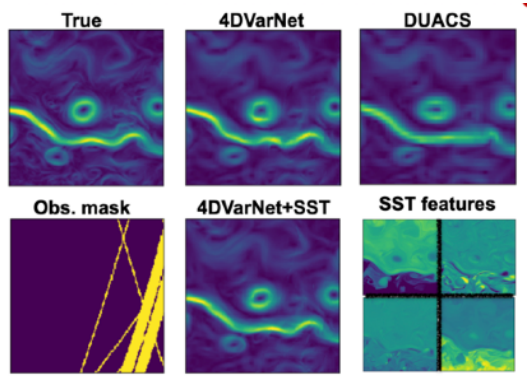
Interpolation & Forecasting



Learning where to sample ?



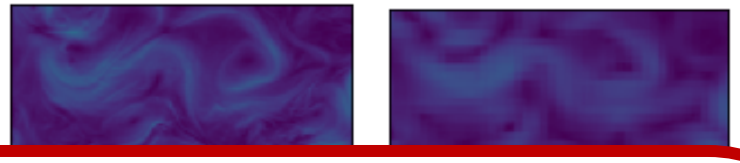
Multimodal learning



Trainable observation operators

Learning where to sample ?

Learning what to measure ?



4DVarNet models with trainable observation models

$$\hat{x} = \arg\min_x \|x - y\|^2 + \lambda \|x - \phi(x)\|^2$$

Sparse sampling operator

$$\|H(z) * (x - y)\|^2$$

$$\text{s.t. } \forall z, \ \|H(z)\|_1 < \epsilon$$

Multimodal observation

$$\|x - y\|^2 + \alpha \|G * x - F * z\|^2$$

Trainable observation operators

4DVarNet models with trainable observation models

Spase sampling operator

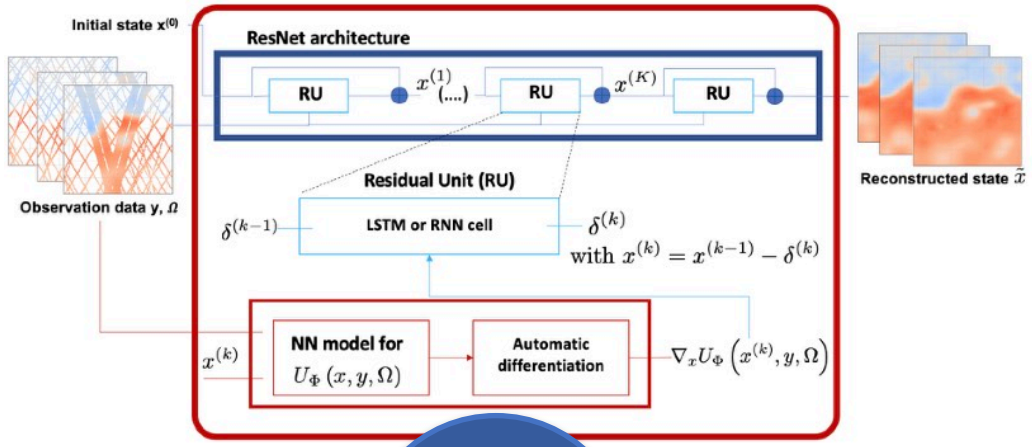
$$\|H(z) * (x - y)\|^2$$

$$\text{s.t. } \forall z, \ \|H(z)\|_1 < \epsilon$$

Multimodal observation

$$\|x - y\|^2 + \alpha \|G * x - F * z\|^2$$

End-to-end 4DVarNet



Supervised training loss

(under sparsity constraint for the optimal sampling case)

Multimodal data assimilation

SSH-SST case-study

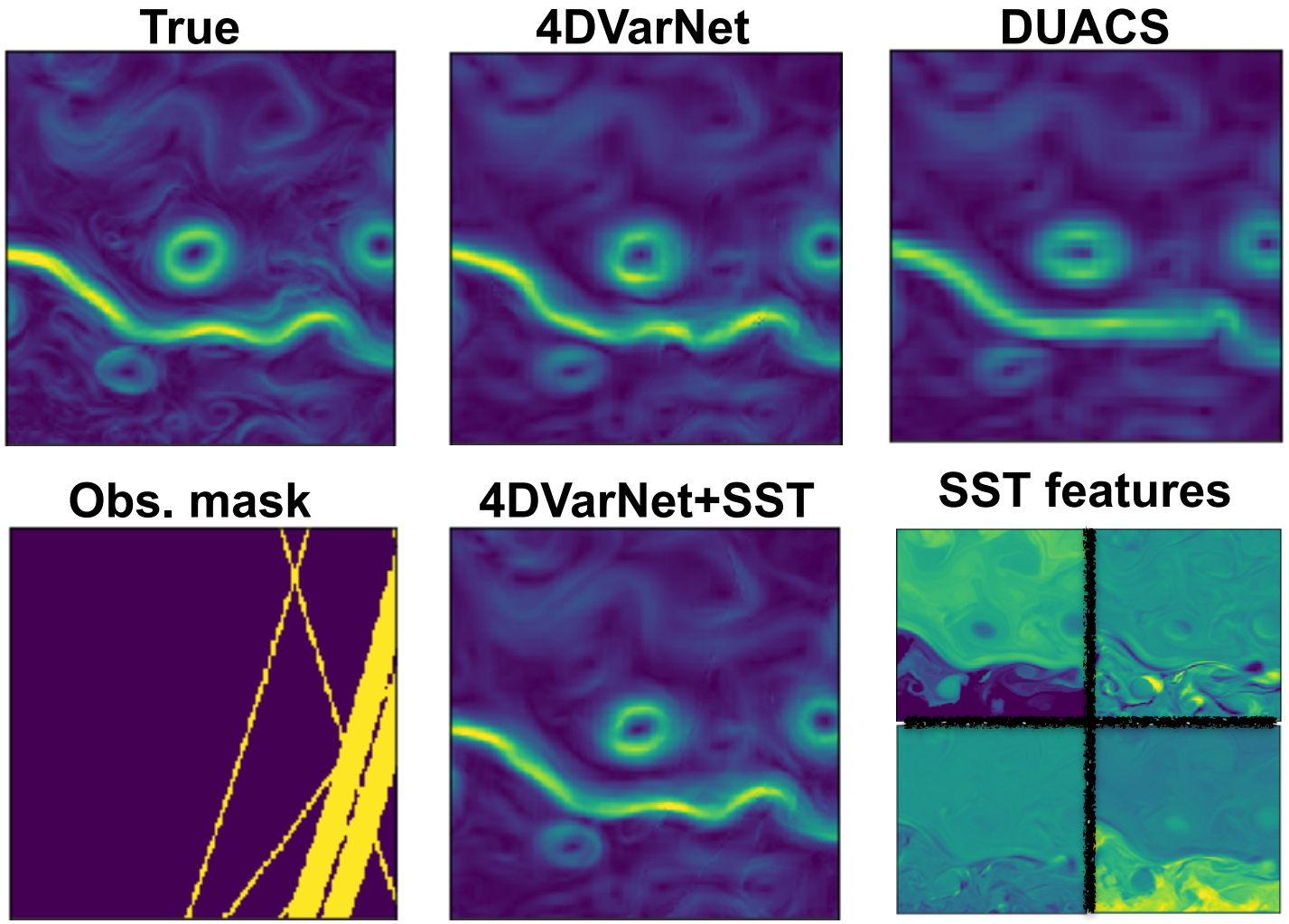
OSSE with NATL60 data

4-nadir-altimeter + SWOT + DUACS baseline

Gulf Stream area (10°x10°)

63% vs. 53% gain in SSH MSE w.r.t. DUACS with/ without SST (Winter period)

50% vs. 13% gain in SSH MSE w.r.t. DUACS using nadir altimeter data only



Optimal sampling

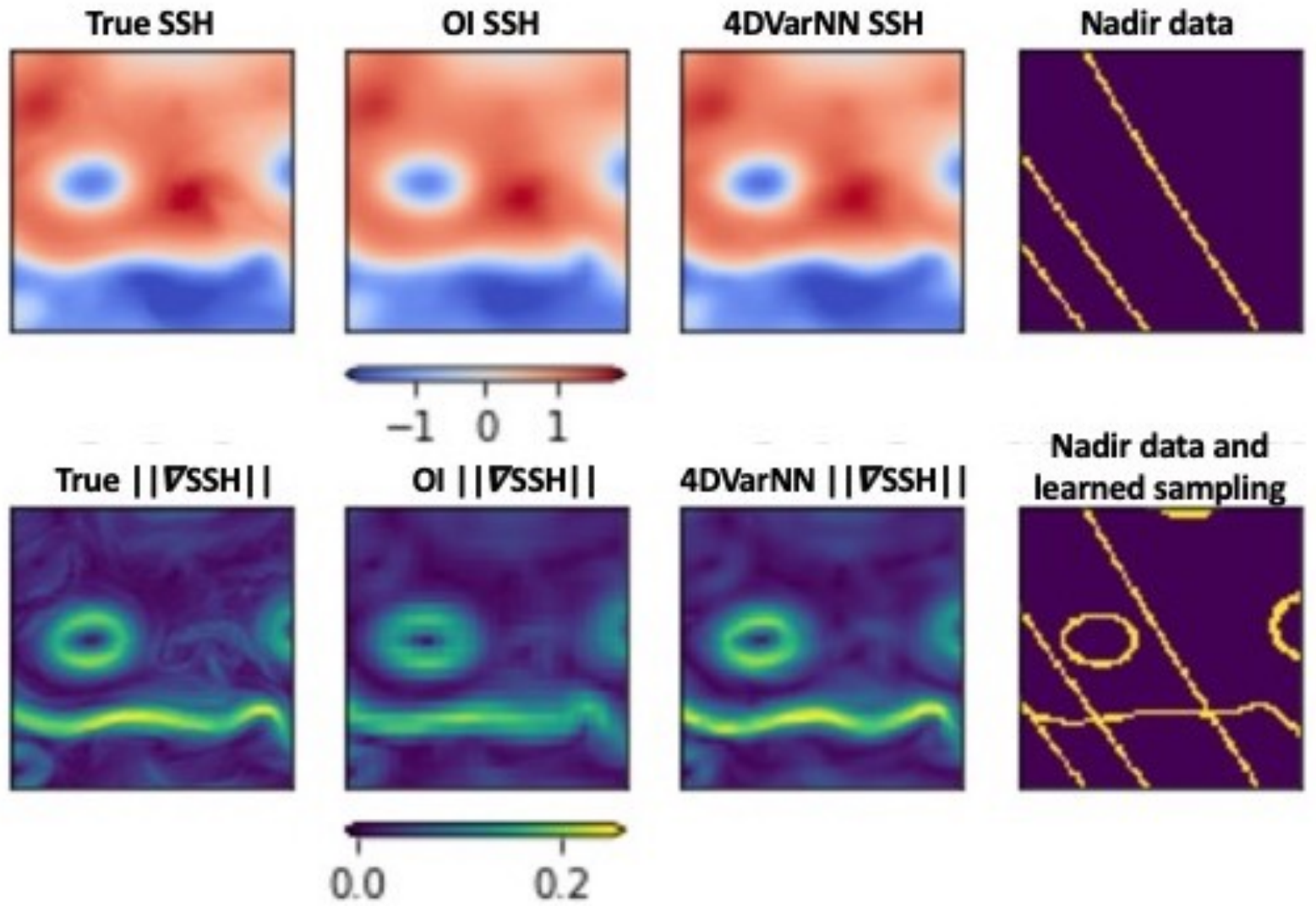
SSH case-study

OSSE with NATL60 data

4-nadir-altimeter + DUACS baseline

Gulf Stream area (10°x10°)

Mean relative gain of 60% in the reconstruction of the SSH using the learned sampling (~6% of the pixels vs. 1.3% for nadir altimeters)

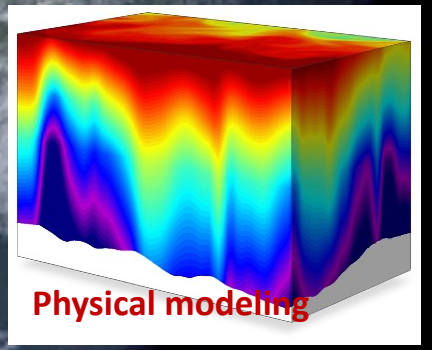


General question

How to solve sampling gaps and extract high-level information for ocean monitoring and surveillance ?

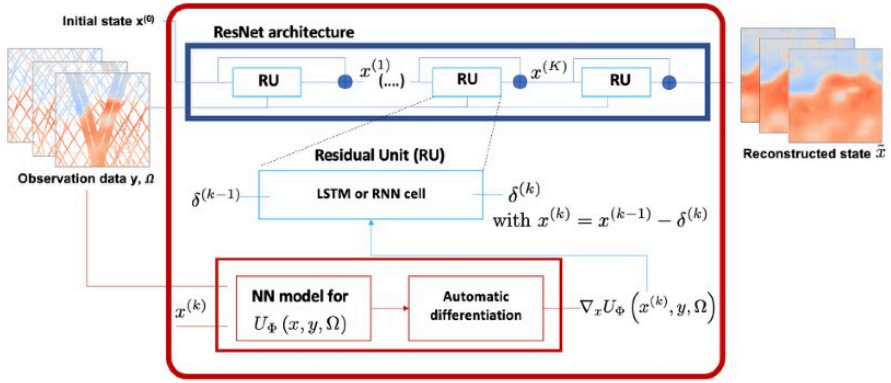
+

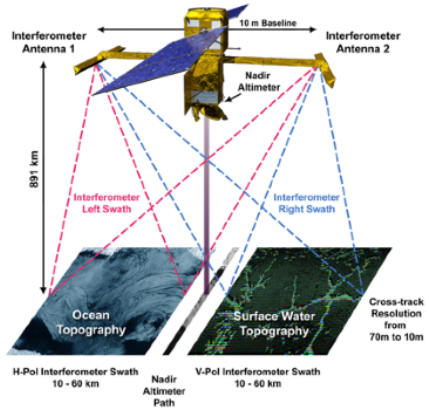
+

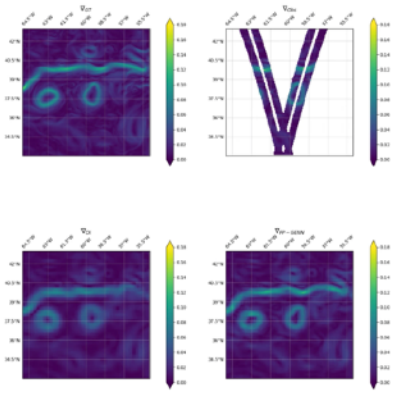


Key messages

- **Physics-informed learning for satellite ocean remote sensing**
- **Trainable variational DA models** (observation model, prior, solver)
- **Application to interpolation, forecasting sampling and multimodal synergies**
- **Generic framework beyond space oceanography**







Preprint: https://arxiv.org/abs/2006.03653
Code: https://github.com/CIA-Oceanix/DinAE_4DVarNN_torch

Thank you.

AI Chair OceaniX 2020-2024

Physics-informed AI for Observation-Driven Ocean AnalytiX

PI: R. Fablet, Prof. IMT Atlantique, Brest

Web: https://cia-oceanix.github.io/

References

- **Meta-learning:**
  - Andrychowicz, et al (2016). Learning to learn by gradient descent by gradient descent. NIPS, 2016
  - Hospedales et al. (2020). Meta-learning in neural networks: A survey. arXiv preprint arXiv:2004.05439.
- **End-to-end learning for Inverse Problems:**
  - Dieleman et al. End-to-end learning for music audio. IEEE ICASSP 2014. doi: 10.1109/ICASSP.2014.6854950
  - Lucas et al. (2018). Using Deep Neural Networks for Inverse Problems in Imaging: Beyond Analytical Methods. IEEE SPM, 35(1), 20–36. doi: 10.1109/MSP.2017.2760358
- **Inverse problems with Plug-and-play priors:**
  - McCann et al. (2017). Convolutional Neural Networks for Inverse Problems in Imaging: A Review. IEEE SPM, 34(6), 85–95. doi: 10.1109/MSP.2017.2739299
- **Deep unfolding methods:**
  - Chen et al. (2015). On Learning Optimized Reaction Diffusion Processes for Effective Image Restoration. In Proc. IEEE CVPR (pp. 5261–5269).
  - Yan et al. Deep ADMM-Net for Compressive Sensing MRI. NIPS, 2016.
  - Fablet et al. Learning Variational Data Assimilation Models and Solvers. JAMES, 2021.