

Learning Generalized Quasi-Geostrophic Models Using Deep Neural Numerical Models

Redouane Lguensat, Julien Le Sommer, Sammy Metref, Emmanuel Cosme, Ronan Fablet

▶ **To cite this version:**

Redouane Lguensat, Julien Le Sommer, Sammy Metref, Emmanuel Cosme, Ronan Fablet. Learning Generalized Quasi-Geostrophic Models Using Deep Neural Numerical Models. NeurIPS 2019 : 33rd Conference on Neural Information Processing Systems, Dec 2019, Vancouver, Canada. hal-02366600

HAL Id: hal-02366600
https://imt-atlantique.hal.science/hal-02366600

Submitted on 16 Nov 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Learning Generalized Quasi-Geostrophic Models Using Deep Neural Numerical Models

Redouane Lguensat, Julien Le Sommer, Sammy Metref, Emmanuel Cosme
Université Grenoble Alpes, CNRS, IRD, Grenoble INP, IGE; 38000 Grenoble, France.
{firstname.lastname}@univ-grenoble-alpes.fr

Ronan Fablet
IMT Atlantique; Lab-STICC, Brest, France
ronan.fablet@imt-atlantique.fr

Abstract

We introduce a new strategy designed to help physicists discover hidden laws governing dynamical systems. We propose to use machine learning automatic differentiation libraries to develop hybrid numerical models that combine components based on prior physical knowledge with components based on neural networks. In these architectures, named Deep Neural Numerical Models (DNNMs), the neural network components are used as building-blocks then deployed for learning hidden variables of underlying physical laws governing dynamical systems. In this paper, we illustrate an application of DNNMs to upper ocean dynamics, more precisely the dynamics of a sea surface tracer, the Sea Surface Height (SSH). We develop an advection-based fully differentiable numerical scheme, where parts of the computations can be replaced with learnable ConvNets, and make connections with the single-layer Quasi-Geostrophic (QG) model, a baseline theory in physical oceanography developed decades ago.

1 Introduction

Physical modeling is still one of the most striking examples where humans are a long way ahead of pure Machine Learning (ML) systems. Recently, numerous research efforts have been directed into designing ML algorithms, especially deep neural networks, that can learn the basic laws of physics from data [1, 2, 3, 4]. These works mostly agree on the importance of interpretability and respect of physical constraints, which is still not straightforward when using black-box regressors such as neural networks [5]. In particular, many works focused on the case where data describing the dynamical system of interest is assumed to be governed by a system of partial differential equations (PDEs) and where a certain physical a priori is known [6, 7, 8, 9]. From the perspective of ocean sciences, many results shown in the aforementioned references were run on toy models, and more investigation is needed on real ocean satellite-derived or model simulation data. Few works set the foot in this direction: in [10] ideas from optical flow video prediction were linked to an advection-diffusion model and considered to forecast Sea Surface Temperature (SST), while in [11, 12] the goal was to infer the dynamics of a latent variable from partial and noisy observations of SST and Sea Level Anomaly (SLA) respectively.

The general idea behind the present work consists in standing on the shoulders of the current understanding of ocean variables by physical oceanographers, and include as much as we can of their knowledge in the design of our NN architecture. In this paper, we demonstrate this strategy on upper ocean dynamics, and more precisely the dynamics of the Sea Surface Height (SSH). We present a fully-differential advection-diffusion architecture which generalizes the Quasi-Geostrophy

Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019), Vancouver, Canada.

(QG) theory, one of the main baselines for forecasting SSH [13]. To ensure numerical stability and stay close to realistic solutions, ideas from numerical schemes were considered in developing the architecture at the expense of depth and memory load.

2 Model

Deep Neural Numerical Models Automatic differentiation (AD) has a long history in the numerical modeling community. In ocean sciences for instance, OpenAD an open source code for AD has been used to calculate adjoints of popular general circulation models such as MITgcm [14, 15]. However, these tools do not account for the training of neural networks straightforwardly, and are in general designed to handle Fortran based codes. We refer by Deep Neural Numerical Models (DNNMs) to fully differentiable numerical models that can incorporate easily trainable NNs. Depending on the complexity of the involved PDE equations, corresponding DNNMs require a considerable amount of technical and engineering work, to rewrite standard numerical model codes into fully differentiable architectures that allows training NNs by backpropagation. This could not be possible without the ongoing flare in the Deep Learning community and in particular in AD librairies such as PyTorch and Tensorflow [16, 17, 18]. DNNMs can range from fully NN-based architectures such as ResNets [19, 20] to complex physical constrained architectures such as in [7, 10, 8]. Here, we propose an advection-diffusion DNNM, that solves the following equations:

$$a) \quad \Phi = \mathcal{T}_1(\Psi); \quad U = \mathcal{T}_2(\Psi); \quad V = \mathcal{T}_3(\Psi); \qquad b) \quad \frac{\partial \Phi}{\partial t} + U\frac{\partial \Phi}{\partial x} + V\frac{\partial \Phi}{\partial y} = D\nabla^2\Phi, \quad (1)$$

where ∇^2 is the 2D Laplacian operator, U and V are components of the nondivergent velocity field, and D the diffusion coefficient. These equations describe the evolution of the flow field Ψ through the advection-diffusion of a proxy variable Φ obtained by a given transformation $\mathcal{T}_1$. In case $\mathcal{T}_1$ is the identity, we fall into classical models as the one studied in [10]. Using PyTorch, we develop a DNNM where the discretization of the PDE involves the use of a 3^{rd} order upwind scheme and a 1^{st} order Euler scheme in time. This scheme is stable as long as the Courant–Friedrichs–Lewy condition (CFL) is satisfied, implying that model integration is done in small steps dt. Ψ at each dt is obtained through the inversion of Equation 1a, for example if $\mathcal{T}_1$ is linear, we use Conjugate Gradient (CG) method with constant boundary conditions. In practice, given a good initial Ψ guess, the CG is stopped after few iterations (less than 5) to permit real-time execution and to avoid computational burden.

QG-Net The 1-layer Quasi-geostrophic (QG) model is a reduced model that describes the evolution of oceanic flows close to geostrophic balance [21]. Recently, this model was found to be a good baseline to dynamically interpolate SSH fields through temporal gaps [13]. To forecast SSH dynamics, the numerical model uses q the Potential Vorticity (PV) as a proxy variable that is advected by Geostrophic Velocities (GV). This process is governed by the following equations [22, 23, 24]:

$$q = \frac{g}{f}(\nabla^2 h - \frac{h}{L_R^2}); \quad U^g = \frac{-g}{f}\frac{\partial h}{\partial y}; \quad V^g = \frac{g}{f}\frac{\partial h}{\partial x}; \quad \frac{\partial q}{\partial t} + U^g\frac{\partial q}{\partial x} + V^g\frac{\partial q}{\partial y} + \beta V^g = 0, \quad (2)$$

where h is the SSH field, g is the gravity constant, f is the Coriolis parameter, L_R is the first Rossby deformation radius. U^g and V^g are the Geostrophic Velocities, and βV^g is a term that accounts for meridional advection of PV. This model is in accordance with the DNNM framework presented above. We find the correspondant $\mathcal{T}_1$, $\mathcal{T}_2$, $\mathcal{T}_3$ and call it QG-Net. Gradients and Laplacians were rewritten into filter convolutions. One CG iteration can already give an acceptable solution if the guess field respects the following pattern:

$$h_{dt}^{guess} = h_0; \quad h_{kdt}^{guess} = 2 * h_{(k-1)dt} - h_{(k-2)dt} \quad k \in \mathbb{N}^* \backslash \{1\}, \quad (3)$$

therefore we used the equations of the CG algorithm [25] to write the equivalent of one CG iteration. The resulting architecture is illustrated in Fig1(a,b). It benefits from high flexibility, since several building blocks can be interchanged with ConvNets, making it a playground for several modeling choices such as the ones exposed in the next section. Yet, due to the high memory cost of using

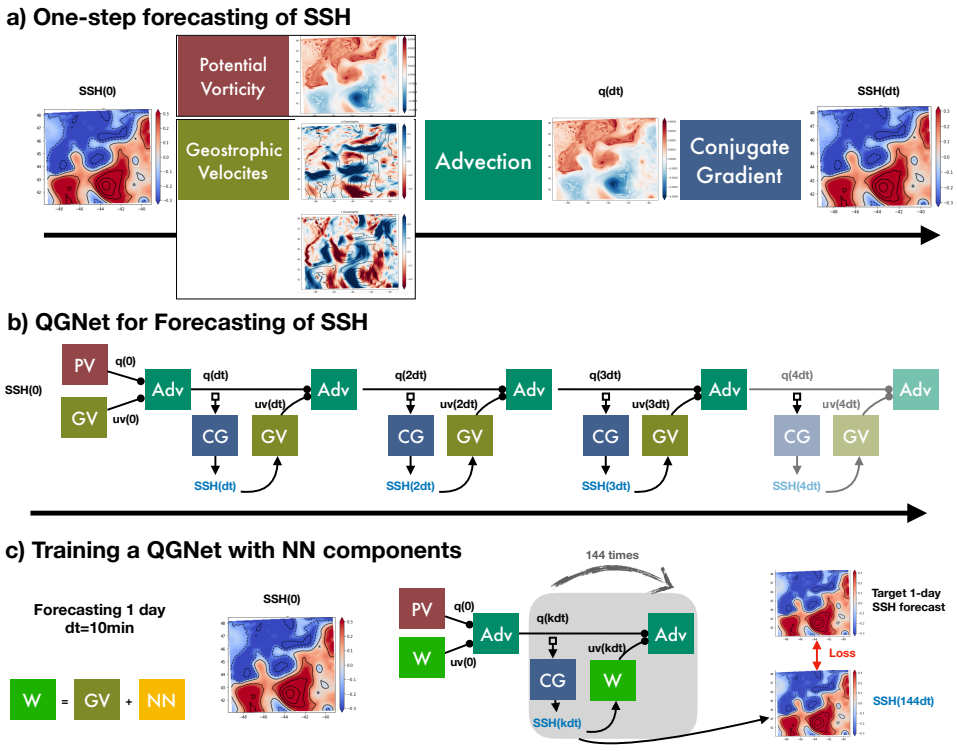


Figure 1: a) 1-step integration of QG equations; b,c) use cases of QG-Net

float64 precision and low values of dt needed for the integration step (which results in a high number of iterations), training QG-Net in an acceptable time requires several high-end Graphics Processing Units (GPUs).

3 Experiments

Data and Experimental details We use NATL60, a dataset from a comprehensive realistic ocean model simulation based on NEMO ocean engine run at kilometric resolution over the North Atlantic basin [26]. Study region is a $10^\circ \times 10^\circ$ box located on the Gulf Stream, a region with challenging physics. Four Nvidia Tesla V100 GPUs were used for the computations described in the experiments.

Discovering insights about hidden laws from data NATL60 ocean circulation model is governed by complex physics not covered entirely by the QG theory. Yet, we want to investigate to which extent QG-Net can reveal the limits of this theory. A simple illustrative example consists in assuming that the PV is advected by some unknown fields U^* and V^* which are first-order derivations of SSH through a linear operator F. Concretely, given that the 2D gradient filter used to calculate the geostrophic velocities V^g and U^g in the original Python code are respectively $F_{QG} = \begin{pmatrix} -0.25 & 0 & 0.25 \\ -0.25 & 0 & 0.25 \end{pmatrix}$ and F_{QG}^T its transpose, we replace F_{QG} in QG-Net by a 6-parameters trainable filter and retrieve the resulting filter from a training procedure using NATL60 data. This resorts to:

$$U^* = -\frac{g}{f}\frac{F^T \circledast SSH}{\partial y} \quad V^* = \frac{g}{f}\frac{F \circledast SSH}{\partial x} \tag{4}$$

We set a 1-day SSH forecasting experiment, and use $dt = 10min$ for QG-Net, meaning that 144 blocks are needed (Fig1 (b,c)). Note that we use shared weights across the blocks. QG-Net in this experiment has then only 6 parameters which are the weights of F. 18 SSH images of size 200×150 are used for training (one each 20 days from June 2012 to June 2013) using the BFGS algorithm and a loss function $\mathcal{L}$ composed of three terms: i) mean square error between the QG-Net 1-day forecast

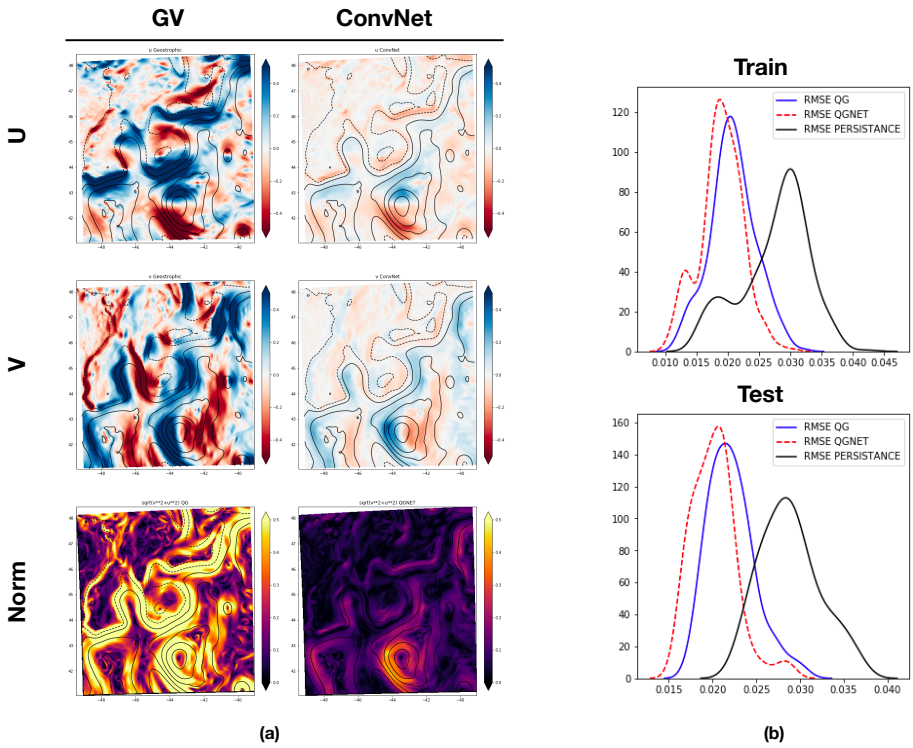


Figure 2: a) An example of the Geostrophic velocities, their norm and the nonlinear additive components produced by the trained ConvNet. SSH contours are shown in black. b) RMSE distributions on the train and test datasets

and the NATL60 target 1-day forecast scaled by the variance of the target, ii) a loss penalizing velocity fields with high divergence, iii) L2-regularization of the weights

$$\mathcal{L} = \frac{1}{n}\Sigma_{i=1}^{n}\left(\frac{SSH_{i,NATL60}^{1day} - SSH_{i,QGNet}^{1day}}{\sigma_i(SSH_{NATL60}^{1day})}\right)^2 + \|\nabla.([U_0^*, V_0^*])\|_2^2 + 10^{-3}\|F\|_2^2 \tag{5}$$

The result of the optimization yields $F = \begin{pmatrix} -0.2629 & 0.0029 & 0.2592 \\ -0.2099 & -0.0008 & 0.2124 \end{pmatrix}$, keep in mind that we are not expecting to find exactly F_{QG} due to the complex dynamics of NATL60. Therefore, from a completely random filter, QG-Net found F which is close to F_{QG}, a proof perhaps that the capacity of this type of models is reached and that the PV is best advected by the GV, as QG-theory claims.

Supplementing known physics with nonlinear learnable components In this experiment, we consider the same 1-day SSH forecast experiment but we assume that at each time step the PV is advected by the GV plus NN a nonlinear transformation of SSH (Fig1(c)). NN is a 2-layer ConvNet with 16 $\{3 \times 3\}$ filters, Batch Normalization and leaky ReLU activations, the output layer is a linear layer with 2 channels that are added to U^g and V^g respectively to form W the new velocity fields. QG-Net has 2545 trainable parameters and we split our data into 122 SSH images of size 200×150 (1 SSH map each 3 days from 14 Jun 2012 to 13 Jun 2013), then after a 10-day gap we take 32 snapshots as our test dataset (1 SSH map each 3 days from 24 Jun to 29 Sep 2013). Our network is trained using Adam optimizer with an initial learning rate of $1e^{-3}$ which is later multiplied by 0.1 each 100 epoch. Batch size is 4 samples distributed on the 4 GPUs cards. The loss function considered here is the scaled mean square error used in Eq.5. To ensure a stable gradient flow at the beginning of the optimization a scalar parameter initialized as zero is multiplied to the ConvNet velocities.

At the end of training, we unplug the ConvNet from QG-Net, resulting in a NN component that takes SSH as input and yields a deterministic "perturbation" of GV. A clear benefit from this setup is that

the trained component can be plugged back in the original Python code and avoid computational load at test time. Fig2(b) presents the RMSE distributions of the standard QG, our QG-Net and a naive constant model (persistance). Adding the ConvNet component to the GV has slightly improved the standard QG model, this is an indication that a nonlinear velocity term can model SSH dynamics beyond the standard QG. Fig2(a) shows an example of the additional velocities produced by the ConvNet along with GV for the same SSH input. We observe that the output of the ConvNet has a significant amplitude along SSH contours (ocean fronts), and that the fields follow a special pattern that depends on GV and are not completely random. Interpreting the ConvNet in this experiment is not straightforward and calls for more investigation to convert it into tangible equations that could be inspected by physical oceanographers.

4 Conclusion

We show that combining deep learning automatic differentiation libraries and numerical models could help designing hybrid models with trainable parameters and represent a test bed to evaluate established physical theories or seek intuition for developing new ones. We believe that this work represents a modest step for helping physicists developing innovative physical models.

Acknowledgments

The authors would like to thank Clément Ubelmann from CLS for the 1-layer QG Python code that can be found here `https://github.com/redouanelg/qgsw-DI`. Most of the computations presented in this paper were performed using the GRICAD infrastructure (`https://gricad.univ-grenoble-alpes.fr`), which is partly supported by the Equip@Meso project (reference ANR-10-EQPX-29-01) of the programme Investissements d'Avenir supervised by the Agence Nationale pour la Recherche.
R. Lguensat is funded through a postdoctoral grant from Centre National d'Etudes Spatiales (CNES), he also acknowledges the support of NVIDIA Corporation under the NVIDIA GPU Grant program. S. Metref is funded by ANR through contract number ANR-17- CE01-0009-01. R. Fablet was supported by Labex Cominlabs (grant SEACS), CNES (grant OSTST-MANATEE) and ANR (EUR Isblue and Melody).

References

[1] Tailin Wu and Max Tegmark. Toward an AI physicist for unsupervised learning. *arXiv preprint arXiv:1810.10525*, 2018.

[2] Raban Iten, Tony Metger, Henrik Wilming, Lídia Del Rio, and Renato Renner. Discovering physical concepts with neural networks. *arXiv preprint arXiv:1807.10300*, 2018.

[3] Sam Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *arXiv preprint arXiv:1906.01563*, 2019.

[4] Ce Wang, Hui Zhai, and Yi-Zhuang You. Emergent schrödinger equation in an introspective machine learning architecture. *Science Bulletin*, 2019.

[5] Giuseppe Carleo, Ignacio Cirac, Kyle Cranmer, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *arXiv preprint arXiv:1903.10563*, 2019.

[6] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[7] Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

[8] Zichao Long, Yiping Lu, and Bin Dong. Pde-net 2.0: Learning pdes from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, page 108925, 2019.

[9] Peter Y Lu, Samuel Kim, and Marin Soljačić. Extracting interpretable physical parameters from spatiotemporal systems using unsupervised learning. *arXiv preprint arXiv:1907.06011*, 2019.

[10] Emmanuel de Bezenac, Arthur Pajot, and Patrick Gallinari. Deep learning for physical processes: Incorporating prior scientific knowledge. *arXiv preprint arXiv:1711.07970*, 2017.

[11] Ibrahim Ayed, Emmanuel de Bézenac, Arthur Pajot, Julien Brajard, and Patrick Gallinari. Learning dynamical systems from partial observations. *arXiv preprint arXiv:1902.11136*, 2019.

[12] Said Ouala, Duong Nguyen, Lucas Drumetz, Bertrand Chapron, Ananda Pascual, Fabrice Collard, Lucile Gaultier, and Ronan Fablet. Learning latent dynamics for partially-observed chaotic systems. *arXiv preprint arXiv:1907.02452*, 2019.

[13] Clement Ubelmann, Patrice Klein, and Lee-Lueng Fu. Dynamic interpolation of sea surface height and potential applications for future high-resolution altimetry mapping. *Journal of Atmospheric and Oceanic Technology*, 32(1):177–184, 2015.

[14] Uwe Naumann, Jean Utke, Carl Wunsch, Chris Hill, P Heimbach, Mike Fagan, Nathan Tallent, and Michelle Strout. Adjoint code by source transformation with openad/f. In *Proceedings of the European Conference on Computational Fluid Dynamics (ECCOMAS CFD 2006). TU Delft*, 2006.

[15] Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch. Openad/f: A modular open-source tool for automatic differentiation of fortran codes. *ACM Transactions on Mathematical Software (TOMS)*, 34(4):18, 2008.

[16] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[17] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[18] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18(153), 2018.

[19] E Weinan. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5(1):1–11, 2017.

[20] François Rousseau, Lucas Drumetz, and Ronan Fablet. Residual networks as flows of diffeomorphisms. *Journal of Mathematical Imaging and Vision*, pages 1–11, 2019.

[21] G. K. Vallis. *Atmospheric and Oceanic Fluid Dynamics*. Cambridge University Press, Cambridge, U.K., 2006.

[22] K Shafer Smith and Geoffrey K Vallis. The scales and equilibration of midocean eddies: Freely evolving flow. *Journal of Physical Oceanography*, 31(2):554–571, 2001.

[23] BL Hua and DB Haidvogel. Numerical simulations of the vertical structure of quasi-geostrophic turbulence. *Journal of the atmospheric sciences*, 43(23):2923–2936, 1986.

[24] Lee-Lueng Fu and Glenn R Flierl. Nonlinear energy and enstrophy transfers in a realistically stratified ocean. *Dynamics of Atmospheres and Oceans*, 4(4):219–246, 1980.

[25] Jonathan Richard Shewchuk et al. An introduction to the conjugate gradient method without the agonizing pain, 1994.

[26] Jean-Marc Molines. meom-configurations/NATL60-CJM165: NATL60 code used for CJM165 experiment, March 2018.