GEOMETRY-ADAPTIVE PHYSICS-INFORMED NEURAL SOLVER FOR PARAMETRIC PARTIAL DIFFERENTIAL E-QUATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Partial differential equations (PDEs) are notoriously difficult to solve, and traditional numerical approximation schemes have high computational costs. Recently, hybrid neural-numerical solvers have been developed to ride on the modern trend of fully end-to-end learning systems. Typical approaches can be divided into physics-informed neural networks (PINNs), which approximately satisfy a given set of PDEs by constraining neural networks, and neural solvers, which learn solutions by training data. However, these solvers cannot fulfil the properties of the generic solver for real-world simulations, which should be fast, stable, accurate, no training data, multi-scale and multi-physics, resolution invariance, geometric adaptation. In this work, we build a geometry-adaptive physics-informed neural solver (GeoPINS) that satisfies these properties, combining the advantages of no training data in PINNs, as well as fast, accurate and resolution-invariant architectures offered by Fourier neural operators (FNO). In particular, to adapt to complex and irregular geometries that exist in the real world, we reformulate PINNs in geometry-adaptive space by taking full advantage of coordinate transformations and the efficiency of numerical methods (including pseudo-spectral and high-order finite difference methods) in solving the spatial gradient. In order to overcome observed issues of conventional PINNs when solving challenging PDEs (such as high-frequency, multi-physics), we rewrite FNO as an efficient visual mixer to construct spatial-temporal representations and validate our approach on many popular PDEs on both regular and irregular domains, demonstrating fast, stable, and accurate performance, as well as resolution-invariant, geometry-adaptive properties for real-world simulations. In addition, GeoPINS achieves superior accuracy compared to previous solvers at different zero-shot super-resolution settings.

1 INTRODUCTION

Solving partial differential equations (PDEs) is of huge importance for real-world simulations (e.g., flood modelling, weather forecasting, or astronomical simulations). Yet in many practical scenarios with complex boundary conditions, such as the Navier-Stokes equations for fluid flow Gal-Chen & Somerville (1975) and shallow water equations for flood inundation modelling Bates et al. (2010), it is not feasible to find analytical solutions. Due to the lack of analytical solutions, their finite-dimensional approximation is resorted to based on traditional numerical approaches, e.g., finite difference (FD), finite volume (FV), finite element (FE), and pseudo-spectral (PS) methods, which have been developed and advanced in the last decades. However, for real-world simulations, traditional numerical solvers often require a large computational effort, especially for complex systems and may not even be feasible for real-time applications Ming et al. (2020). In addition, traditional methods (e.g. FD) are highly dependent on the resolution: coarse grids are fast but low accurate; fine grids are accurate but slow. For example, a global flood solver based on FD has a resolution of up to 1 km, but this requires a dramatic increase in computational resources Zhou et al. (2021). *A "good" PDE solver for real-world simulations* should satisfy the following basic conditions: i) Fast and accurate. Stable and accurate approximation schemes can be obtained with minimal computational overhead; ii) Resolution-invariant. The solver can efficiently represent dynamic processes at different spatial

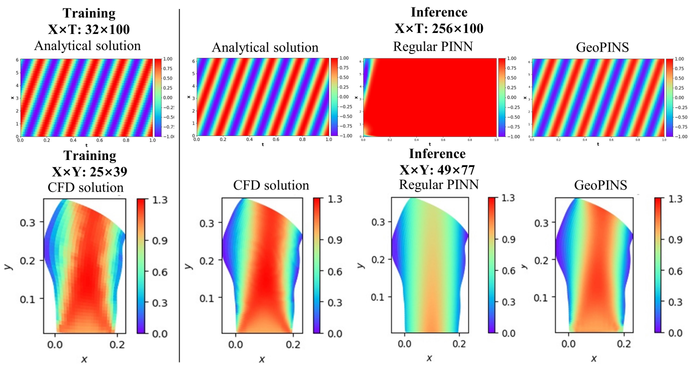


Figure 1: 1-D convection equations on regular domain (first row): Train on a spatial-temporary resolution 32×100 directly infer on a spatial-temporary resolution 256×100; 2-D steady incompressible Navier-Stokes equations on irregular domain (second row): Train on a spatial resolution 25×39 directly infer on a spatial resolution 49×77. Analytical solution and computational fluid dynamics (CFD) represent ground truth. The comparison between regular PINN based on fully-connected NNs (third column) and GeoPINS (fourth column).

and temporal resolutions; iii) Geometry-adaptive. The solver can be adapted to general scientific problems that exist in the real world, where the geometry is often complex and irregular.

Motivated by this, there has been a recent interest in developing machine learning (ML) approaches to find solutions for the underlying PDEs (and/or work in tandem with numerical solutions). Recently, deep neural networks (DNNs) have been demonstrated to be promising for solving PDEs or fluid dynamics systems Cai et al. (2022); Karniadakis et al. (2021); Lu et al. (2021). A recent work involves physics-informed neural networks (PINNs) Raissi et al. (2019) by approximating solution functions and neural operators Anandkumar et al. (2020); Li et al. (2020); Brandstetter et al. (2022); Li et al. (2022) by learning solution operators. Currently in NN-based ML methods, PINNs are mainly defined in a pointwise way. Despite their noticeable empirical success, they may fail to solve challenging PDEs when the solution exhibits high-frequency components Krishnapriyan et al. (2021); Wang et al. (2021b); Fuks & Tchelepi (2020). For example, as reported in Fig. 1, one can clearly see that conventional PINN is unable to learn the solution of 1-D convection equations on a regular domain with convection coefficient equal to 30. Furthermore, according to the regular PINN's solution of 2-D steady incompressible Navier-Stokes equations in Fig. 1, the fully-connected networks struggle to learn multiphysics (conservation of energy, momentum and mass) simultaneously. Furthermore, FNO is a resolution-invariant (zero-shot super-resolution) operator not only in the spatial domain but also in the temporal domain, because the Fourier representation space $\mathbb{R}^{d_r}$ does not depend on the spatiotemporal resolution of the physical domain Ω. Moreover, due to the fast Fourier transformation (FFT), FNO is a fast architecture that allows to obtain accurate results for PDEs. However, FNO carries out supervised learning on a given dataset, which is infeasible for real-world simulation. Importantly, due to the existence of the convolution operation, FNO is only able to deal with problems defined on regular (rectangular) domains with uniform grids. For complex and irregular geometries, FNO is no longer valid.

Bearing these concerns in mind, aiming to construct a "good" PDE solver for real-world simulations without using any labeled data, we propose a geometry-adaptive physics-informed neural solver (GeoPINS), that overcomes the shortcomings of both PINNs and neural operators. Compared to PINNs, GeoPINS has a better spatial-temporal representation space, so, GeoPINS converges faster and more accurately. It achieves geometry adaptation and resolution invariance when solving PDEs. Our contributions can be summarized as follows: (1) We propose the GeoPINS, that combines the advantages of leveraging PINNs to overcome the need for training data and leveraging neural operators. Notably, GeoPINS can train on a lower spatio-temporal resolution and directly infer at higher resolution, as shown in Fig. 1, which provides an effective way to refine solutions of PDEs over large areas. (2) GeoPINS is a dynamic geometry-adaptive solver, which can adapt to complex and irregular geometries that exist in the real world. Geometry-adaptive physics-constrained (GeoPC) loss is constructed by reformulating PINNs in geometry-adaptive space. Different spatial-temporal gradients (including gradients on the computational domain and gradients of coordinate transformations) can be solved by a proposed numerical scheme with high precision and high speed. (3) We demonstrate FNO is an effective spatial-temporal model to resolve issues of conventional

PINNs when solving challenging PDEs (such as high-frequency, multiphysics). Importantly, we find FNO is an effective vision mixer. Furthermore, we analyze that Fourier feature embeddings by FNO can make networks more suitable for modeling PDEs' systems in low dimensions. The details are provided in §C in Appendix. (4) We evaluate our method by solving the 1-D Convection equation and 1-D Burgers' equation on regular domains, 2-D incompressible Navier-Stokes equation on irregular domains, and 2-D Shallow water equation on the surface of sphere. Experiments show that GeoPINS is fast, stable, and accurate while retaining excellent resolution-invariant (zero-shot super-resolution), geometry-adaptive properties for real-world simulations. Additionally, GeoPINS achieves higher accuracy compared to previous solvers under different zero-shot super-resolution setting.

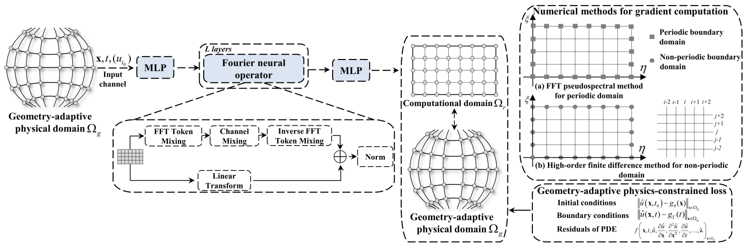
2 Geometry-adaptive physics-informed neural solver

2.1 Overview

The Geometry-adaptive physics-informed neural solver (GeoPINS) is built with input channels I composed of coordinates $\mathbf{x}$, time domain t and available initial condition u_{t_0} of the physical domain Ω_g. As shown in Fig. 2, the input is first lifted to a higher dimensional representation v_0 by multi-layer perceptrons (MLP). Then several FNO layers defined in Li et al. (2020) (K_0,K_1,...,K_l) are applied to extract efficient spatial-temporal representations v_l, where kernel function $\mathcal{K}_\phi$: $\mathbb{R}^{d\times d} \to \mathbb{R}^{d_r\times d_r}$ is a neural network parameterized by ϕ. d represents the dimensionality of discretization in the domain Ω. Notably, different from FNO layers in Li et al. (2020), different dimensions of representation space d_r in different FNO layers are used to adapt different PDE problems with lower computational parameters. The outputs $\hat{u}(t)$ (e.g. velocity or pressure) in computational domain Ω_r is obtained by applying MLP projection on v_l. The forward process can be formulated as,

$$v_l = (K_l \circ \sigma_l \circ \cdots \circ \sigma_1 \circ K_0) \circ MLP(I), \quad \hat{u}(t)\|_{\mathbf{x}\in\Omega_r} = MLP(v_l). \tag{1}$$

Here $\hat{u}(\mathbf{x}, t)\|_{\mathbf{x}\in\Omega_r}$ and $\hat{u}(\mathbf{x}, t)\|_{\mathbf{x}\in\Omega_g}$ are equal at corresponding nodes. The neural solver is trained with a proposed GeoPC loss constructed by reformulating PINNs in geometry-adaptive space, which is described in detail below.



$\mathbf{x}, t, (u_{t_0})$
Input channel
MLP
L layers
Fourier neural operator
MLP
Geometry-adaptive physical domain Ω_g
FFT Token Mixing
Channel Mixing
Inverse FFT Token Mixing
Norm
Linear Transform
Computational domain Ω_r
Geometry-adaptive physical domain Ω_g
Numerical methods for gradient computation
ξ
η
Periodic boundary domain
Non-periodic boundary domain
(a) FFT pseudospectral method for periodic domain
i-2 i-1 i i+1 i+2
j+2 j+1 j j-1 j-2
(b) High-order finite difference method for non-periodic domain
Geometry-adaptive physics-constrained loss
Initial conditions $\|\hat{u}(\mathbf{x},t_0)-g_0(\mathbf{x})\|_{\mathbf{x}\in\Omega_g}$
Boundary conditions $\|\hat{u}(\mathbf{x},t)-g_\Gamma(t)\|_{\mathbf{x}\in\Omega_g}$
Residuals of PDE $f\left(\mathbf{x},t,\hat{u},\frac{\partial\hat{u}}{\partial\mathbf{x}},\frac{\partial^2\hat{u}}{\partial\mathbf{x}^2},\frac{\partial\hat{u}}{\partial t},\ldots,\lambda\right)_{\mathbf{x}\in\Omega_g}$

Figure 2: The full architecture of Geometry-adaptive physics-informed neural solver.

2.2 Reformulating PINNs in geometry-adaptive space

Given the geometry-adaptive physical domains $\Omega_g = [x, y]$ and computational domains $\Omega_r = [\eta, \xi]$, the transformation $\mathcal{G}$ between coordinates of the geometry-adaptive physical domain (Ω_g) and regular computational domain (Ω_r) can be defined as,

$$x = \mathcal{G}(\eta, \xi), \quad y = \mathcal{G}(\eta, \xi). \tag{2}$$

To enable physics-constrained learning in the geometry-adaptive space, the original PDEs defined in the physical domain has to be recast as the form in computational domain. Specifically, the first and second derivatives in Ω_g are transformed into Ω_r as,

$$\begin{aligned}
\frac{\partial}{\partial x} &= \left(\frac{\partial}{\partial \eta}\right)\left(\frac{\partial \eta}{\partial x}\right) + \left(\frac{\partial}{\partial \xi}\right)\left(\frac{\partial \xi}{\partial x}\right), \quad \frac{\partial}{\partial y} = \left(\frac{\partial}{\partial \eta}\right)\left(\frac{\partial \eta}{\partial y}\right) + \left(\frac{\partial}{\partial \xi}\right)\left(\frac{\partial \xi}{\partial y}\right) \\
\frac{\partial^2}{\partial x^2} &= \frac{\partial \eta}{\partial x}\frac{\partial \eta}{\partial x}\frac{\partial^2}{\partial \eta^2} + \frac{\partial \eta}{\partial x}\frac{\partial \xi}{\partial x}\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial \xi}{\partial x}\frac{\partial \xi}{\partial x}\frac{\partial^2}{\partial \xi^2} + \left[\frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi}\left(\frac{\partial \eta}{\partial x}\right) + \frac{\partial \eta}{\partial x}\frac{\partial}{\partial \eta}\left(\frac{\partial \eta}{\partial x}\right)\right]\frac{\partial}{\partial \eta} \\
\frac{\partial^2}{\partial y^2} &= \frac{\partial \eta}{\partial y}\frac{\partial \eta}{\partial y}\frac{\partial^2}{\partial \eta^2} + \frac{\partial \eta}{\partial y}\frac{\partial \xi}{\partial y}\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial \xi}{\partial y}\frac{\partial \xi}{\partial y}\frac{\partial^2}{\partial \xi^2} + \left[\frac{\partial \xi}{\partial y}\frac{\partial}{\partial \xi}\left(\frac{\partial \eta}{\partial y}\right) + \frac{\partial \eta}{\partial y}\frac{\partial}{\partial \eta}\left(\frac{\partial \eta}{\partial y}\right)\right]\frac{\partial}{\partial \eta}.
\end{aligned} \tag{3}$$

With these modified derivative terms, the GeoPC loss is defined as,

$$\begin{aligned}
\mathcal{L}_{\text{GeoPC}} &= \min_\theta \|\mathcal{L}_{\text{data}}\|_{\Omega_g} + \lambda \|\mathcal{L}_{PDE}\|_{\Omega_g} \\
\mathcal{L}_{\text{data}} &= \|\mathcal{L}_{IC}\|_{\Omega_g} + \|\mathcal{L}_{BC}\|_{\Omega_g} \\
\mathcal{L}_{PDE} &= f\left(\mathbf{x}, t, \hat{u}, \frac{\partial \hat{u}}{\partial \mathbf{x}}, \frac{\partial^2 \hat{u}}{\partial \mathbf{x^2}}, \frac{\partial \hat{u}}{\partial t}, \ldots, \vartheta\right), \quad \mathbf{x} \in \Omega_g, t \in [0, T],
\end{aligned} \tag{4}$$

where λ is a regularization parameter that controls the emphasis on residuals of PDE. $\mathcal{L}_{\text{data}}$ (including $\mathcal{L}_{IC}$/$\mathcal{L}_{BC}$) measure the mismatch between the NN prediction and the initial/boundary conditions in Eq. 11, and $\mathcal{L}_{PDE}$ is a constraint on the residual of the PDE system. θ denotes the NN parameters.

Two types of gradients in GeoPC loss, including gradients on regular computational domain (e.g. $\frac{\partial}{\partial \eta}$, $\frac{\partial}{\partial \xi}$) and gradients of coordinate transformations (e.g. $\frac{\partial \eta}{\partial x}$, $\frac{\partial \eta}{\partial y}$, $\frac{\partial \xi}{\partial x}$, $\frac{\partial \xi}{\partial y}$), need to be solved efficiently. The most general way in PINNs Raissi et al. (2019); Jin et al. (2021) to compute the exact gradient is to use the auto-differentiation library of neural networks (autograd). The autograd method is a pointwise operation, which is general and exact. However, it is usually slow and memory-consuming on the space domain. The gradient solving based on numerical methods are significantly more effective than autograd, since the number of neural operator parameters is usually much greater than the grid size. Thus, FFT pseudo-spectral (FFT PS) numerical methods Boyd (2001) and high-order finite difference (FD) numerical methods Li et al. (1995) are utilized to calculate spatial gradients of periodic and non-periodic domains, respectively, which is shown in the upper right corner of Fig. 2. The high-order FD methods are used to calculate temporal gradients. Notably, the detailed derivation and explanation of gradients on computational domain and gradients of coordinate transformations are presented in §A and §B in Appendix, respectively.

3 NUMERICAL EXPERIMENTS

In this section, we conduct empirical experiments to examine the efficacy of the proposed GeoPINS by solving the 1-D Convection equation and 1-D Burgers' equation on regular domains, 2-D incompressible Navier-Stokes equation on irregular domains, and 2-D Shallow water equation on the surface of sphere. Importantly, the detailed implementation details are shown in §D in Appendix. State-of-the-art solvers based on PINNs are considered as benchmarks, including regular PINNs Krishnapriyan et al. (2021), Physics-informed geometry-adaptive convolutional neural networks (PhyGeoNet) Gao et al. (2021a), PINO Li et al. (2021), and PINN-DeepONet Wang et al. (2021b).

3.1 PDEs ON REGULAR DOMAIN

We first consider a 1-D convection problem to verify the proposed GeoPINS, which is a common hyperbolic wave equation. The detail forms, implementation, results and visualization of GeoPINS can be found in §E.1 in Appendix. Second, in order to compare with various advanced neural-based solvers Li et al. (2020; 2021); Wang et al. (2021b), the 1-D Burgers equation on regular domain is utilized. The 1-D Burgers equation is a non-linear PDE with periodic boundary conditions. The implementation detail, results and analysis are included in §E.2 in Appendix. Through a large number of experiments on regular geometries, it has been demonstrated that GeoPINS can achieve significant performance improvement compared with other advanced neural-based solvers.

3.2 PDEs ON IRREGULAR DOMAIN

We consider the 2-d steady incompressible Navier-Stokes equations on irregular river (§E.3 in Appendix) and the shallow water equation on the sphere in latitudelongitude coordinates (§E.4 in Appendix). Results on the Navier-Stokes equation in §E.3 show that GeoPINS provides a favorable opportunity for fast and refined-resolution solutions of large-scale PDEs through the geometry-adaptive and resolution-invariant properties Results on the geostrophic flow test case in §E.4 show that GeoPINS is a prominent (fast and accurate) dynamic geometry-adaptive solver for global dynamics simulation.

4 CONCLUSION

We have observed that regular PINNs and neural operators have fundamental limitiations for solving PDEs in real-world simulations. Thus, a generic solver, GeoPINS, is proposed with excellent resolution-invariant (zero-shot super-resolution), geometry-adaptive properties. GeoPINS achieves superior accuracy compared to PINNs-based solvers on many popular PDEs relevant to real-world problems. A shortcoming of our current work is inaccuracies of solving gradients for coarse meshes by finite differences. Hence, we should look in the future towards learned gradients that allow an efficient and accurate computation.

REFERENCES

Anima Anandkumar, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Nikola Kovachki, Zongyi Li, Burigede Liu, and Andrew Stuart. Neural operator: Graph kernel network for partial differential equations. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

Paul D Bates, Matthew S Horritt, and Timothy J Fewtrell. A simple inertial formulation of the shallow water equations for efficient two-dimensional flood inundation modelling. *Journal of Hydrology*, 387(1-2):33–45, 2010.

Alex Bihlo and Roman O Popovych. Physics-informed neural networks for the shallow-water equations on the sphere. *Journal of Computational Physics*, 456:111024, 2022.

John P Boyd. *Chebyshev and Fourier spectral methods*. Courier Corporation, 2001.

Johannes Brandstetter, Daniel Worrall, and Max Welling. Message passing neural pde solvers. *arXiv preprint arXiv:2202.03376*, 2022.

Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, pp. 1–12, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

A Farcy and T Alziary de Roquefort. Chebyshev pseudospectral solution of the incompressible navier-stokes equations in curvilinear domains. *Computers & fluids*, 16(4):459–473, 1988.

Bengt Fornberg. The pseudospectral method: Comparisons with finite differences for the elastic wave equation. *Geophysics*, 52(4):483–501, 1987.

Olga Fuks and Hamdi A Tchelepi. Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. *Journal of Machine Learning for Modeling and Computing*, 1(1), 2020.

Tzvi Gal-Chen and Richard CJ Somerville. On the use of a coordinate transformation for the solution of the navier-stokes equations. *Journal of Computational Physics*, 17(2):209–228, 1975.

Han Gao, Luning Sun, and Jian-Xun Wang. Phygeonet: physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state pdes on irregular domain. *Journal of Computational Physics*, 428:110079, 2021a.

Peng Gao, Jiasen Lu, Hongsheng Li, Roozbeh Mottaghi, and Aniruddha Kembhavi. Container: Context aggregation network. *arXiv preprint arXiv:2106.01401*, 2021b.

Ronald L Gilliland. Solutions of the shallow water equations on a sphere. *Journal of Computational Physics*, 43(1):79–94, 1981.

John Guibas, Morteza Mardani, Zongyi Li, Andrew Tao, Anima Anandkumar, and Bryan Catanzaro. Adaptive fourier neural operators: Efficient token mixers for transformers. *arXiv preprint arXiv:2111.13587*, 2021.

Thomas Y Hou and Ruo Li. Computing nearly singular solutions using pseudo-spectral methods. *Journal of Computational Physics*, 226(1):379–397, 2007.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Hrvoje Jasak, Aleksandar Jemcov, Zeljko Tukovic, et al. Openfoam: A c++ library for complex physics simulations. In *International workshop on coupled methods in numerical dynamics*, volume 1000, pp. 1–20. IUC Dubrovnik Croatia, 2007.

Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Georgios Kissas, Yibo Yang, Eileen Hwuang, Walter R Witschey, John A Detre, and Paris Perdikaris. Machine learning in cardiovascular flows modeling: Predicting arterial blood pressure from non-invasive 4d flow mri data using physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 358:112623, 2020.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34, 2021.

Ming Li, Tao Tang, and Bengt Fornberg. A compact fourth-order finite difference scheme for the steady incompressible navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 20(10):1137–1151, 1995.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, Anima Anandkumar, et al. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2020.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *arXiv preprint arXiv:2111.03794*, 2021.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *arXiv preprint arXiv:2207.05209*, 2022.

Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021.

Xiaodong Ming, Qiuhua Liang, Xilin Xia, Dingmin Li, and Hayley J Fowler. Real-time flood forecasting based on a high-performance 2-d hydrodynamic model and numerical weather predictions. *Water Resources Research*, 56(7):e2019WR025583, 2020.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Basri Ronen, David Jacobs, Yoni Kasten, and Shira Kritchman. The convergence rate of neural networks for learned functions of different frequencies. *Advances in Neural Information Processing Systems*, 32, 2019.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34, 2021.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021a.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Science advances*, 7(40):eabi8605, 2021b.

David L Williamson, John B Drake, James J Hack, Rüdiger Jakob, and Paul N Swarztrauber. A standard test set for numerical approximations to the shallow water equations in spherical geometry. *Journal of computational physics*, 102(1):211–224, 1992.

Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. *arXiv preprint arXiv:2111.11418*, 2021.

Xudong Zhou, Catherine Prigent, and Dai Yamazaki. Toward improved comparisons between land-surface-water-area estimates from a global river model and satellite observations. *Water Resources Research*, 57(5):e2020WR029256, 2021.

A Gradients on the computational domain.

The principal advantage of FFT PS methods is the great accuracy of the resulting discretization scheme for a given number of nodes, which is the limit of finite-difference methods of increasing orders Fornberg (1987). In addition, the saving of computational resources is another advantage for FFT PS methods. Applying the PS method amounts to performing a periodic discrete convolution, requiring two fast Fourier transforms. Specifically, a FFT of the predictions (velocity, pressure, depth and others) $\mathcal{F}(\hat{u})$ in computational domain is firstly used to compute the discrete Fourier space. Then, the solution of gradients can be obtained in the Fourier space. Finally, the inverse FFT $\mathcal{F}^{-1}$ is utilized to construct the full solution in computational domain. Thus,

$$\begin{aligned}
\frac{\partial \hat{u}}{\partial \eta} &= \mathcal{F}^{-1}\left(ik_\eta \mathcal{F}(\hat{u})\right), \quad \frac{\partial^2 \hat{u}}{\partial \eta^2} = \mathcal{F}^{-1}\left(-k_\eta^2 \mathcal{F}(\hat{u})\right), \quad k_\eta = \frac{2\pi w_\eta}{L_\eta} \\
\frac{\partial \hat{u}}{\partial \xi} &= \mathcal{F}^{-1}\left(ik_\xi \mathcal{F}(\hat{u})\right), \quad \frac{\partial^2 \hat{u}}{\partial \xi^2} = \mathcal{F}^{-1}\left(-k_\xi^2 \mathcal{F}(\hat{u})\right), \quad k_\xi = \frac{2\pi w_\xi}{L_\xi} \\
w_\eta &= \{-N_\eta/2, -N_\eta/2+1, \ldots, N_\eta/2-1\} \quad w_\xi = \{-N_\xi/2, -N_\xi/2+1, \ldots, N_\xi/2-1\},
\end{aligned} \tag{5}$$

where k_η and k_ξ represent η-direction wave-numbers and ξ-direction wave-numbers, respectively. The spatial grid size of the computational domain is $N_\eta \times N_\xi$. L_η and L_ξ are spatial lengths of η-direction and ξ-direction, respectively. Importantly, if rapidly-oscillating PDE functions with relatively few grid points are solved (e.g. Convection equation with spatial size 32×32), an effect known as aliasing will be observed Boyd (2001). Aliasing means that wavenumbers of magnitude greater than $k_\eta/2$ or $k_\xi/2$ are incorrectly represented as lower wavenumbers. A 2/3 filter Hou & Li (2007) is used to avoid aliasing instability. Filtering means removing the mode that leads to aliasing, which can be done by damping the high wavenumbers when computing the gridents.

However, FFT PS methods are restricted to computation in rectangular domains with periodic boundary conditions. On the contrary, some PS solutions are designed for non-periodic boundary conditions. For example, the non-periodic boundary (e.g. no-slip wall condition) can be handled by Chebyshev PS methods Farcy & de Roquefort (1988). However, a non-uniform collocation grid (e.g. cosine function) needs to be built for Chebyshev polynomials, which makes FD-based gradient computation of coordinate transformations unstable and introduces construction problems of the differential scheme. Therefore, a four-order difference scheme is constructed to obtain stable and accurate gradients on the computational domain with a non-periodic boundary. Specifically, the first derivatives are approximated by four-order central differences,

$$\begin{aligned}
\frac{\partial \hat{u}}{\partial \eta} &\approx \frac{-\hat{u}_{\eta+2,\xi} + 8\hat{u}_{\eta+1,\xi} - 8\hat{u}_{\eta-1,\xi} + \hat{u}_{\eta-2,\xi}}{12\delta_\eta} + O\left((\delta_\eta)^4\right) \\
\frac{\partial \hat{u}}{\partial \xi} &\approx \frac{-\hat{u}_{\eta,\xi+2} + 8\hat{u}_{\eta,\xi+1} - 8\hat{u}_{\eta,\xi-1} + \hat{u}_{\eta,\xi-2}}{12\delta_\xi} + O\left((\delta_\xi)^4\right),
\end{aligned} \tag{6}$$

where δ_η and δ_ξ are spatial steps of η-direction and ξ-direction, respectively. The four-order difference can be expressed by an convolution filter in the algorithm. For the boundary, third-order one-sided differences (i.e., upwind/downwind) are applied to stabilize the difference scheme.

B GRADIENTS OF COORDINATE TRANSFORMATIONS.

The gradient solutions of coordinate transformations $\mathcal{G} : \Omega_r \mapsto \Omega_g$ can be divided into two cases. Firstly, when analytical forms of $\mathcal{G}$ are available, gradients of coordinate transformations can be solved exactly. For example, in terms of solving the Convection equation and Burgers' equation in regular rectangular grid ($\Omega_r = \Omega_g$), gradients of coordinate transformations are expressed as,

$$\begin{bmatrix} \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \\ \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{7}$$

For solving the shallow water equation on the surface of a sphere Bihlo & Popovych (2022); Gilliland (1981), given the Spherical domains $\Omega_g = [\theta, \phi]$ (θ=colatitude,ϕ=longitude) and Cartesian computational domains $\Omega_r = [\eta, \xi]$, gradients of $\mathcal{G}$ are expressed as (a is the radius of the earth),

$$\begin{bmatrix} \frac{\partial \eta}{\partial \theta} & \frac{\partial \eta}{\partial \phi} \\ \frac{\partial \xi}{\partial \theta} & \frac{\partial \xi}{\partial \phi} \end{bmatrix} = \begin{bmatrix} a\cos\theta\cos\phi & -a\sin\theta\sin\phi \\ a\cos\theta\sin\phi & a\sin\theta\cos\phi \end{bmatrix}. \tag{8}$$

Secondly, in most cases, analytical forms of $\mathcal{G}$ are not available, which have to be approximated numerically (e.g. solving incompressible Navier-Stokes equations on an irregular domain). Inspired by Gao et al. (2021a), elliptic coordinate transformation is utilized in this case. For example, $\mathcal{G}$ can be obtained by solving a diffusion equation, considering the one-to-one mapping boundary condition. Formulated as,

$$\begin{aligned} &\alpha\frac{\partial^2 x}{\partial \eta^2} - 2\beta\frac{\partial^2 x}{\partial \eta \partial \xi} + \gamma\frac{\partial^2 x}{\partial \xi^2} = 0, \quad \alpha\frac{\partial^2 y}{\partial \eta^2} - 2\beta\frac{\partial^2 y}{\partial \eta \partial \xi} + \gamma\frac{\partial^2 y}{\partial \xi^2} = 0 \\ &\alpha = \left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2, \gamma = \left(\frac{\partial x}{\partial \eta}\right)^2 + \left(\frac{\partial y}{\partial \eta}\right)^2, \beta = \frac{\partial x}{\partial \eta}\frac{\partial x}{\partial \xi} + \frac{\partial y}{\partial \eta}\frac{\partial y}{\partial \xi} \\ &Boundary : \mathcal{G}(\eta, \xi) = \partial\Omega_g^i \quad for \quad \forall \eta, \xi \in \partial\Omega_r^i, \quad i = 1, \cdots, 4. \end{aligned} \tag{9}$$

By solving Eq. 9 based on four-order difference numerically (Eq. 6), the Jacobian $\frac{\partial y}{\partial \eta}$, $\frac{\partial y}{\partial \xi}$, $\frac{\partial x}{\partial \eta}$ and $\frac{\partial x}{\partial \xi}$ of $\mathcal{G}$ can be obtained. Then, the Jacobian matrix ($J = \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta}$) is applied to modify Eq. 3 (such as first derivatives) as,

$$\begin{aligned} \frac{\partial}{\partial x} &= J^{-1}\left[\left(\frac{\partial}{\partial \eta}\right)\left(\frac{\partial y}{\partial \xi}\right) - \left(\frac{\partial}{\partial \xi}\right)\left(\frac{\partial y}{\partial \eta}\right)\right] \\ \frac{\partial}{\partial y} &= J^{-1}\left[\left(\frac{\partial}{\partial \xi}\right)\left(\frac{\partial x}{\partial \eta}\right) - \left(\frac{\partial}{\partial \eta}\right)\left(\frac{\partial x}{\partial \xi}\right)\right]. \end{aligned} \tag{10}$$

The derivation of second derivatives is presented as follows.

The derivatives of reference domains $\Omega_r = [\eta, \xi]$ with respect to geometry-adaptive physical domains $\Omega_g = [x, y]$ can be expressed as,

$$\begin{aligned} \frac{\partial \eta}{\partial x} &= \frac{1}{J}\frac{\partial y}{\partial \xi} \\ \frac{\partial \xi}{\partial x} &= -\frac{1}{J}\frac{\partial y}{\partial \eta} \\ \frac{\partial \eta}{\partial y} &= -\frac{1}{J}\frac{\partial x}{\partial \xi} \\ \frac{\partial \xi}{\partial y} &= \frac{1}{J}\frac{\partial x}{\partial \eta}, \end{aligned} \tag{11}$$

where the Jacobian matrix $J = \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta}$.

The second derivatives in Ω_g are transformed into Ω_r as,

$$\begin{aligned} \frac{\partial^2}{\partial x^2} &= \frac{\partial \eta}{\partial x}\frac{\partial \eta}{\partial x}\frac{\partial^2}{\partial \eta^2} + \frac{\partial \eta}{\partial x}\frac{\partial \xi}{\partial x}\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial \xi}{\partial x}\frac{\partial \xi}{\partial x}\frac{\partial^2}{\partial \xi^2} + \left[\frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi}\left(\frac{\partial \eta}{\partial x}\right) + \frac{\partial \eta}{\partial x}\frac{\partial}{\partial \eta}\left(\frac{\partial \eta}{\partial x}\right)\right]\frac{\partial}{\partial \eta} \\ \frac{\partial^2}{\partial y^2} &= \frac{\partial \eta}{\partial y}\frac{\partial \eta}{\partial y}\frac{\partial^2}{\partial \eta^2} + \frac{\partial \eta}{\partial y}\frac{\partial \xi}{\partial y}\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial \xi}{\partial y}\frac{\partial \xi}{\partial y}\frac{\partial^2}{\partial \xi^2} + \left[\frac{\partial \xi}{\partial y}\frac{\partial}{\partial \xi}\left(\frac{\partial \eta}{\partial y}\right) + \frac{\partial \eta}{\partial y}\frac{\partial}{\partial \eta}\left(\frac{\partial \eta}{\partial y}\right)\right]\frac{\partial}{\partial \eta}. \end{aligned} \tag{12}$$

Based on Eq. 11 and Eq. 12, the second derivatives are calculated as,

$$\begin{aligned}
\frac{\partial^2}{\partial x^2} &= J^{-2}\left[\frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \xi}\frac{\partial^2}{\partial \eta^2} - \frac{\partial y}{\partial \xi}\frac{\partial y}{\partial \eta}\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial y}{\partial \eta}\frac{\partial y}{\partial \eta}\frac{\partial^2}{\partial \xi^2} + \left[-\frac{\partial y}{\partial \eta}\frac{\partial}{\partial \xi}\left(\frac{\partial y}{\partial \xi}\right) + \frac{\partial y}{\partial \xi}\frac{\partial}{\partial \eta}\left(\frac{\partial y}{\partial \xi}\right)\right]\frac{\partial}{\partial \eta}\right] \\
\frac{\partial^2}{\partial y^2} &= J^{-2}\left[\frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \xi}\frac{\partial^2}{\partial \eta^2} - \frac{\partial x}{\partial \xi}\frac{\partial x}{\partial \eta}\frac{\partial^2}{\partial \eta \partial \xi} + \frac{\partial x}{\partial \eta}\frac{\partial x}{\partial \eta}\frac{\partial^2}{\partial \xi^2} + \left[-\frac{\partial x}{\partial \eta}\frac{\partial}{\partial \xi}\left(\frac{\partial x}{\partial \xi}\right) + \frac{\partial x}{\partial \xi}\frac{\partial}{\partial \eta}\left(\frac{\partial x}{\partial \xi}\right)\right]\frac{\partial}{\partial \eta}\right].
\end{aligned} \tag{13}$$

C Spatial-temporal model

FNO is an effective spatial-temporal model to resolve the issues of PINNs (toy examples are shown in Fig. 1). The discrete FNO is defined by replacing the kernel function in Li et al. (2020) with,

$$\left(\mathcal{K}(\phi)v_t\right)(x) = \mathcal{F}^{-1}\left(R_\phi \cdot (\mathcal{F}v_t)\right)(x) \quad \forall x \in \Omega, \tag{14}$$

where $\mathcal{F}$ and $\mathcal{F}^{-1}$ represent the FFT and inverse FFT, respectively. R_ϕ is a parametric function $\mathbb{R}^d \times \mathbb{R}^{d_r} \rightarrow \mathbb{R}^{d_r} \times \mathbb{R}^{d_r}$ that maps to the values of the appropriate Fourier modes k. Two perspectives are given to theoretically analyze the effectiveness of FNO.

Fourier feature embeddings. FNO may enable faster convergence to high-frequency components of PDEs. Conventional FCNN in PINNs can yield a significantly lower convergence rate for high frequency components of PDEs Wang et al. (2021a); Ronen et al. (2019). However, compared with FCNN, Fourier feature embeddings by FFT in Eq. 14 can make networks better suited for modeling PDEs' systems in low dimensions, thereby overcoming the high-frequency failure problems of PINNs, which can be demonstrated by neural tangent kernel theory Tancik et al. (2020); Jacot et al. (2018).

FNO is an effective vision mixer. Vision transformers (ViT) have recently shown promise in producing rich contextual representations for spatial-temporal tasks Guibas et al. (2021); Dosovitskiy et al. (2020); Yu et al. (2021); Gao et al. (2021b). The general architecture of ViT Dosovitskiy et al. (2020) is a line of models with a stack of vision mixers, which include token mixing $\rightarrow$ channel mixing $\rightarrow$ token mixing. Mixer components can be various attention-based operations Dosovitskiy et al. (2020), spatial MLP Tolstikhin et al. (2021), pooling operation Yu et al. (2021), and others. A key component for the effectiveness of transformers is attributed to proper mixing of tokens.

The FNO in Eq. 14 is a member of vision mixers. Specifically, the FNO can be decomposed into token mixing $\rightarrow$ channel mixing $\rightarrow$ token mixing. Firstly, the FFT $\mathcal{F}v_t$ can be considered the token mixing. Then, the matrix multiplication R_ϕ performs mixing on channel. Finally, the inverse FFT $\mathcal{F}^{-1}$ is a token mixing layer. Detailed process is shown in Fig. 2. Thus, FNO has superior spatiotemporal expressiveness and generalization for challenging PDEs.

D Implementation details.

Four FNO layers are stacked with different Fourier modes k = {12, 12, 9, 9}, channel combinations of the representation space d_r = {16, 24, 24, 32, 32} and GeLU activation for all cases. A layer normalization is used after the FNO layer. We use Adam optimizer Kingma & Ba (2014) with an initial learning rate of 0.001. The hyperparameter λ=1 in GeoPC loss for all cases. $L2$ relative error between the predicted solutions and the analytical/computational fluid dynamics (CFD) solutions is regarded as evaluation measure. For all cases, we run models at least five times with different preset random seeds, and average the relative errors. All the computation is carried on a single Nvidia RTX 3090 GPU with 24GB memory.

E Experiments

E.1 Convection equation

We first consider a 1-D convection problem to verify the proposed GeoPINS. It takes the form,

$$\begin{aligned}
\frac{\partial u}{\partial t} + \beta\frac{\partial u}{\partial x} &= 0, \quad x \in [0, 2\pi], t \in [0, T] \\
u(x, 0) &= h(x),
\end{aligned} \tag{15}$$

Table 1: Results of Convection equations. $A \rightarrow B$ represents training on a spatial resolution with A and directly evaluating on a spatial resolution with B. The values in **bold** are the best.

Methods	Mode Parameters	β=20				β=30				β=40			
		32→256	64→256	128→256	256→256	32→256	64→256	128→256	256→256	32→256	64→256	128→256	256→256
Regular PINN (a 4-layer fully-connected NN)	30701	1.6667	1.7081	1.5183	1.1276	1.7325	1.9008	1.5630	1.0372	1.5085	1.8210	1.0184	1.0993
Regular PINN (a 8-layer fully-connected NN)	1124401	1.2277	1.4874	0.9537	1.3654	0.9569	1.4404	1.3859	1.0723	1.4620	1.1529	1.1012	1.2013
GeoPINS with FD	1141091	0.3128	0.0974	**0.0271**	0.0770	0.4323	0.1308	**0.0332**	0.1077	0.5553	**0.1660**	**0.0524**	**0.2310**
GeoPINS with FFT PS	1141091	**0.0622**	**0.0381**	0.0304	**0.0225**	**0.1608**	**0.1211**	0.1035	**0.0952**	**0.4668**	0.4436	0.4296	0.4215

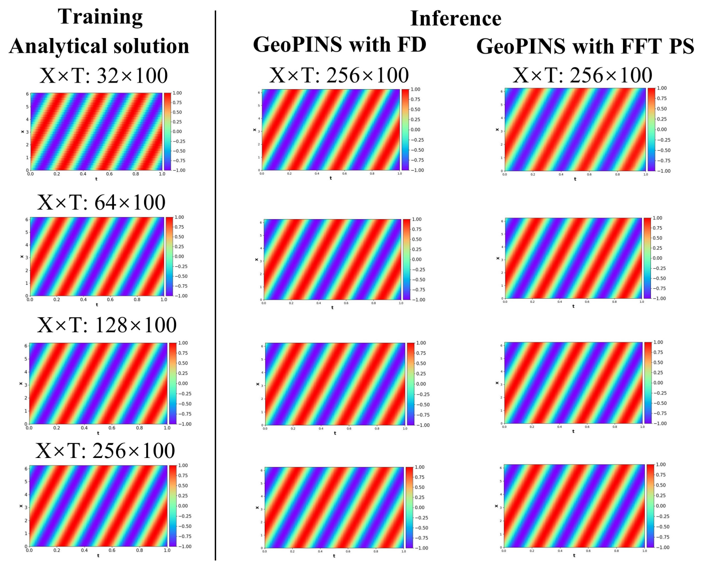


Figure 3: 1-D convection equations on regular domain (β=20): Train on a low spatial-temporary resolution (first column) directly infer on a spatial-temporary resolution 256×100 by GeoPINS with FD (second column) and GeoPINS with FFT PS (third column).

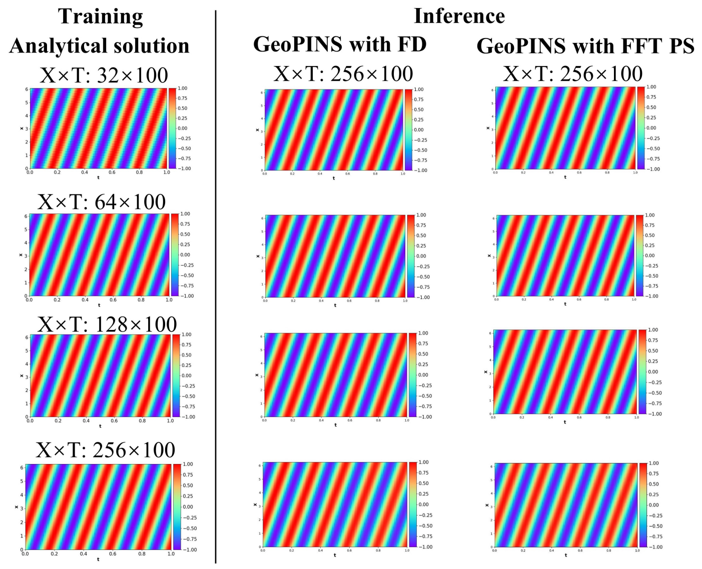


Figure 4: 1-D convection equations on regular domain (β=30): Train on a low spatial-temporary resolution (first column) directly infer on a spatial-temporary resolution 256×100 by GeoPINS with FD (second column) and GeoPINS with FFT PS (third column).

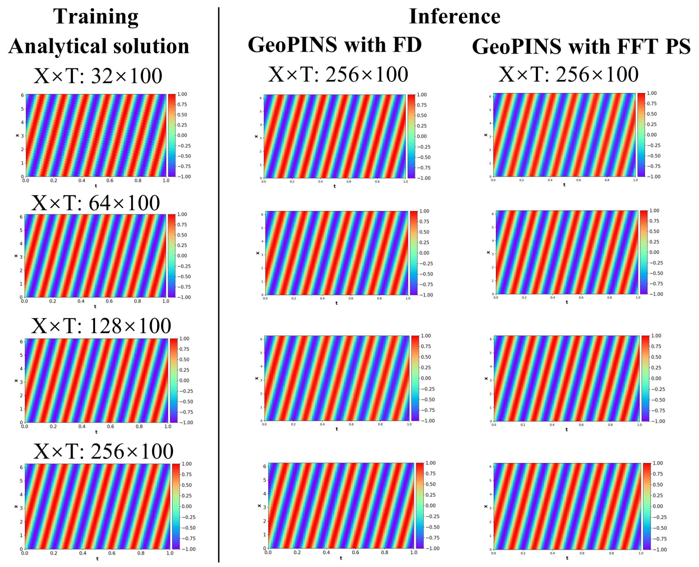


Figure 5: 1-D convection equations on regular domain (β=40): Train on a low spatial-temporary resolution (first column) directly infer on a spatial-temporary resolution 256×100 by GeoPINS with FD (second column) and GeoPINS with FFT PS (third column).

with periodic boundary conditions $u(0, t) = u(2\pi, t)$. Where β is the convection coefficient and $h(x)$ is the initial condition. $h(x) = \sin(x)$ in our experiments. An analytical solution of this problem is used to verify the prediction of neural-based solvers,

$$u_{\text{analytical}}(x, t) = \mathcal{F}^{-1}\left(\mathcal{F}(h(x))e^{-i\beta kt}\right), \tag{16}$$

where k denotes frequency in the Fourier domain. The GeoPC loss is defined by combining Eq. 15 and Eq. 7,

$$\mathcal{L}_{\text{GeoPC}} = \min_{\theta} \|\hat{u}(x, 0) - u(x, 0)\| + \|\hat{u}(0, t) - \hat{u}(2\pi, t)\| + \lambda\|\frac{\partial \hat{u}}{\partial t} + \beta\frac{\partial \hat{u}}{\partial x}^2\|, x \in [0, 2\pi], t \in [0, T]. \tag{17}$$

We use batch size = 1. Adam optimizer is utilized with the learning rate 0.001 that decays by half every 50 epochs. 20000 epochs in total. Each epoch has only one iteration step. The input channel is set to 3, including coordinate x, time t, and initial condition u_0.

The results of our experiments are shown in Table 1. As shown in Table 1, GeoPINS with FD or FFT PS results in a much more accurate solution than the regular PINN. With GeoPINS, the relative error is almost two orders of magnitude lower under different zero-shot super-resolution settings. Furthermore, the error is almost invariant for GeoPINS with different resolutions. Compared with GeoPINS with FD, GeoPINS with FFT PS is generally better performing for this problem with periodic boundary, especially for extreme zero-shot super-resolution (training on spatial-temporary resolution 32×100 and directly inferring on 256×100). However, GeoPINS with FD is more robust for solving complex and non-symmetric convection equations (e.g. β=40). Visualize results of different convection coefficients (β=20,30,40) are shown in Fig. 3, Fig. 4, and Fig. 5. Based on these visualize results, it has been demonstrated that GeoPINS can do zero-shot super-resolution in the spatial domain, and the results are accuracy (similar with analytical solutions) under different convection coefficients.

E.2 Burgers' equation.

The 1-D Burgers equation is formulated as,

$$\begin{aligned}\frac{\partial u}{\partial t} + \frac{\partial\left(u^2/2\right)}{\partial x} &= \nu\frac{\partial^2 u}{\partial x^2}, \quad x \in (0, 1), t \in (0, 1] \\ u(x, 0) &= u_0(x), \quad x \in (0, 1),\end{aligned} \tag{18}$$

where $u_0 \in L^2_{\text{per}}((0,1);\mathbb{R})$ is the initial condition and $\nu = 0.01$ is the viscosity coefficient.

The GeoPC loss can be obtained by combing Burgers equation and gradients of coordinate transformations in regular rectangular grid,

$$\begin{aligned}\mathcal{L}_{\text{GeoPC}} = \min_{\theta} &\|\hat{u}(x,0) - u(x,0)\| + \|\hat{u}(0,t) - \hat{u}(1,t)\| \\ &+ \lambda\|\frac{\partial \hat{u}}{\partial t} + \frac{\partial\left(\hat{u}^2/2\right)}{\partial x} - \nu\frac{\partial^2 \hat{u}}{\partial x^2}\|, \quad x \in (0,1), t \in (0,1].\end{aligned} \tag{19}$$

In order to compare with benchmarks Wang et al. (2021b); Li et al. (2021), we use the same problem setting and training data with PINO Li et al. (2021). Specifically, batch size = 20. 800 instances are used as training data and 200 instances are considered as testing data. We use Adam optimizer with the learning rate 0.001 that decays by half every epoch. 500 epochs in total. The input channel includes coordinate x, time t, and initial condition $u(x,0)$.

Compared to other PDE-constrained operators, GeoPINS is more expressive and thereby achieves better accuracy. Specifically, PINN-DeepONet achieves 0.0138 Wang et al. (2021b). PINO achieves 0.0037 Li et al. (2021). Our GeoPINS with FD achieves 0.0040 and GeoPINS with FFT PS achieves 0.0036. Thus, GeoPINS with FFT PS shows significantly improved performance in PDEs with periodic domains.

E.3 Navier-Stokes equation.

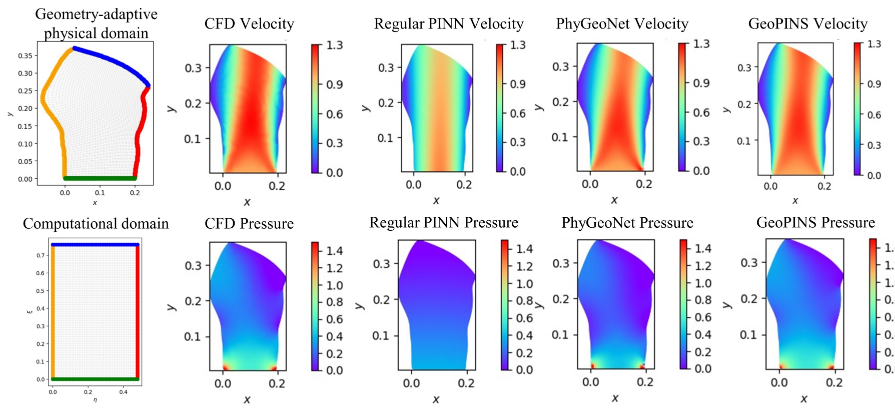


Figure 6: The physical domain (upper left) and reference domain (lower left) for 2-D Navier-Stokes equations. Velocity (the first row) and pressure (the second row) contours of CFD benchmakrk, Regular PINN, PhyGeoNet, and GeoPINS. All methods are fully trained at $\nu = 0.01$.

Table 2: Results of 2-D incompressible Navier-Stokes equations. The values in **bold** are the best. It is demonstrated that GeoPINS has excellent geometry-adaptive and resolution-invariant properties.

Methods	25×39 → 49×77			49×77→ 49×77		
	Training time per epoch	Velocity	Pressure	Training time per epoch	Velocity	Pressure
Regular PINN	0.0313	0.2864	0.6866	0.1571	0.2375	0.6326
PhyGeoNet	0.0615	0.2391	0.6551	0.0640	0.0858	0.3400
GeoPINS	0.0648	**0.0993**	**0.3681**	0.0669	**0.0611**	**0.2769**

We consider the 2-d steady incompressible Navier-Stokes equations,

$$\nabla \cdot \mathbf{v} = 0, \quad (\mathbf{v} \cdot \nabla)\mathbf{v} = -\nabla p + \nabla \cdot (\nu \nabla \mathbf{v}) \tag{20}$$

where $\mathbf{v}$, p represent the velocity vector and pressure, respectively. ν is fluid viscosity. The fluid flow is solved on a geometry-adaptive physical domain Ω_g, as shown in top left of Fig. 6. Similar with Gao et al. (2021a), the no-slip wall boundary condition is used on the left and right boundaries, the boundary condition of inlet (green) is given by $v = [0, 1]$. The boundary condition of outlet (blue) is set as symmetrical boundary and $p = 0$. The computational domain Ω_r is a rectangular grid with step = 0.01 (bottom left in Fig. 6).

The GeoPC loss can be obtained by combing Eq. 20 and Eq. 10,

$$\mathcal{L}_{\text{GeoPC}} = \min_{\theta} \|\mathcal{L}_{\text{boundary}}\|_{\Omega_g} + \lambda \|\nabla \cdot \hat{\mathbf{v}} + (\hat{\mathbf{v}} \cdot \nabla)\hat{\mathbf{v}} + \nabla \hat{p} - \nabla \cdot (\nu \nabla \hat{\mathbf{v}})\|_{\Omega_g} \tag{21}$$

In terms of implementation details of Navier-Stokes equation, batch size = 1, total epochs = 20000. We use Adam optimizer with the learning rate 0.001 that decays by half every epoch. To evaluate the learning performance, the ground truth (same with PhyGeoNet Gao et al. (2021a)) is generated by finite volume (FV)-based numerical simulations. The FV simulations are performed on an open-source FV platform, OpenFOAM Jasak et al. (2007). The input channel is set to 2, including x-component coordinate and y-component coordinate.

The visual results of our experiments are shown in Fig. 6, where the CFD results are used as a benchmark. The regular PINN fails to accurately predict the velocity and pressure distributions, which shows the flaws of PINNs in multiphysics problems. Both the PhyGeoNet and GeoPINS with FD can capture the flow pattern. However, discrepancies are observed in the velocity outlet region and in the high-pressure region in the left and right corners. The GeoPINS prediction result is more accurate and agrees with the CFD benchmark very well. The relative errors for velocity and pressure are listed in Table 2, GeoPINS achieves robust and best performances under different zero-shot super-resolution settings. Contrary to the failure of PhyGeoNet in the setting 25×39 → 49×77, GeoPINS retains the resolution-invariant properties under similar training time as PhyGeoNet. In addition, training time is shorter at lower resolutions. Thus, GeoPINS provides a favorable opportunity for fast and refined-resolution solutions of large-scale PDEs through the geometry-adaptive and resolution-invariant properties.

E.4 Shallow water equation.

Table 3: Quantitative analysis for the nonlinear zonal geostrophic flow test case using GeoPINS with FD and GeoPINS with FFT PS. Reported are the relative errors on the final time t = 5 days.

Methods	Velocity	Height
GeoPINS with FD	0.3770	0.0812
GeoPINS with FFT PS	0.3540	0.0769

The shallow water equation is often solved on a sphere in global atmosphere or ocean models. The shallow-water equations on the sphere in latitudelongitude coordinates $[\theta, \phi]$ are,

$$\begin{aligned}
&\frac{\partial u}{\partial t} + \frac{u}{a\cos\phi}\frac{\partial u}{\partial \theta} + \frac{v}{a}\frac{\partial u}{\partial \phi} - \left(f + \frac{u}{a}\tan\phi\right)v + \frac{g}{a\cos\phi}\frac{\partial h}{\partial \theta} = 0 \\
&\frac{\partial v}{\partial t} + \frac{u}{a\cos\phi}\frac{\partial v}{\partial \theta} + \frac{v}{a}\frac{\partial v}{\partial \phi} + \left(f + \frac{u}{a}\tan\phi\right)u + \frac{g}{a}\frac{\partial h}{\partial \phi} = 0 \\
&\frac{\partial h^*}{\partial t} + \frac{u}{a\cos\phi}\frac{\partial h^*}{\partial \theta} + \frac{v}{a}\frac{\partial h^*}{\partial \phi} + \frac{h^*}{a\cos\phi}\left(\frac{\partial u}{\partial \theta} + \frac{\partial v\cos\phi}{\partial \phi}\right) = 0,
\end{aligned} \tag{22}$$

where u and v are the θcomponents and ϕcomponents of velocity vectors, respectively. $h = h^* + h_b$ is the total water height, with h_b being the height of bottom topography. f is the Coriolis parameter.

Importantly, it is essential to non-dimensionalize the shallow water equation before training GeoPIN-S Kissas et al. (2020); Bihlo & Popovych (2022). Thus, we rewrite 2-D nonlinear shallow water equations as the non-dimensional system,

$$\begin{aligned}
&\frac{\partial \hat{u}}{\partial t} + \frac{\hat{u}}{\cos\hat{\phi}}\frac{\partial \hat{u}}{\partial \hat{\theta}} + \hat{v}\frac{\partial \hat{u}}{\partial \hat{\phi}} - (\hat{f} + \hat{u}\tan\hat{\phi})\hat{v} + \frac{c}{\cos\hat{\phi}}\frac{\partial \hat{h}}{\partial \hat{\theta}} = 0 \\
&\frac{\partial \hat{v}}{\partial t} + \frac{\hat{u}}{\cos\hat{\phi}}\frac{\partial \hat{v}}{\partial \hat{\theta}} + \hat{v}\frac{\partial \hat{v}}{\partial \hat{\phi}} + (\hat{f} + \hat{u}\tan\hat{\phi})\hat{u} + c\frac{\partial \hat{h}}{\partial \hat{\phi}} = 0 \\
&\frac{\partial \hat{h}^*}{\partial t} + \frac{\hat{u}}{\cos\hat{\phi}}\frac{\partial \hat{h}^*}{\partial \hat{\theta}} + \hat{v}\frac{\partial \hat{h}^*}{\partial \hat{\phi}} + \frac{\hat{h}^*}{\cos\hat{\phi}}\left(\frac{\partial \hat{u}}{\partial \hat{\theta}} + \frac{\partial(\hat{v}\cos\hat{\phi})}{\partial \hat{\phi}}\right) = 0,
\end{aligned} \tag{23}$$

where $c = 1$ is an arbitrary non-dimensional constant for global nonlinear zonal geostrophic flow. the re-scaled variables $t = T\hat{t}$, $\hat{\theta} = \theta$, $\hat{\phi} = \phi$, $\hat{u} = \frac{u}{U}$, $\hat{v} = \frac{v}{U}$, $\hat{h}^* = \frac{h^*}{H}$, $\hat{h} = \frac{h}{H}$, and $\hat{f} = Tf$. T, U, and

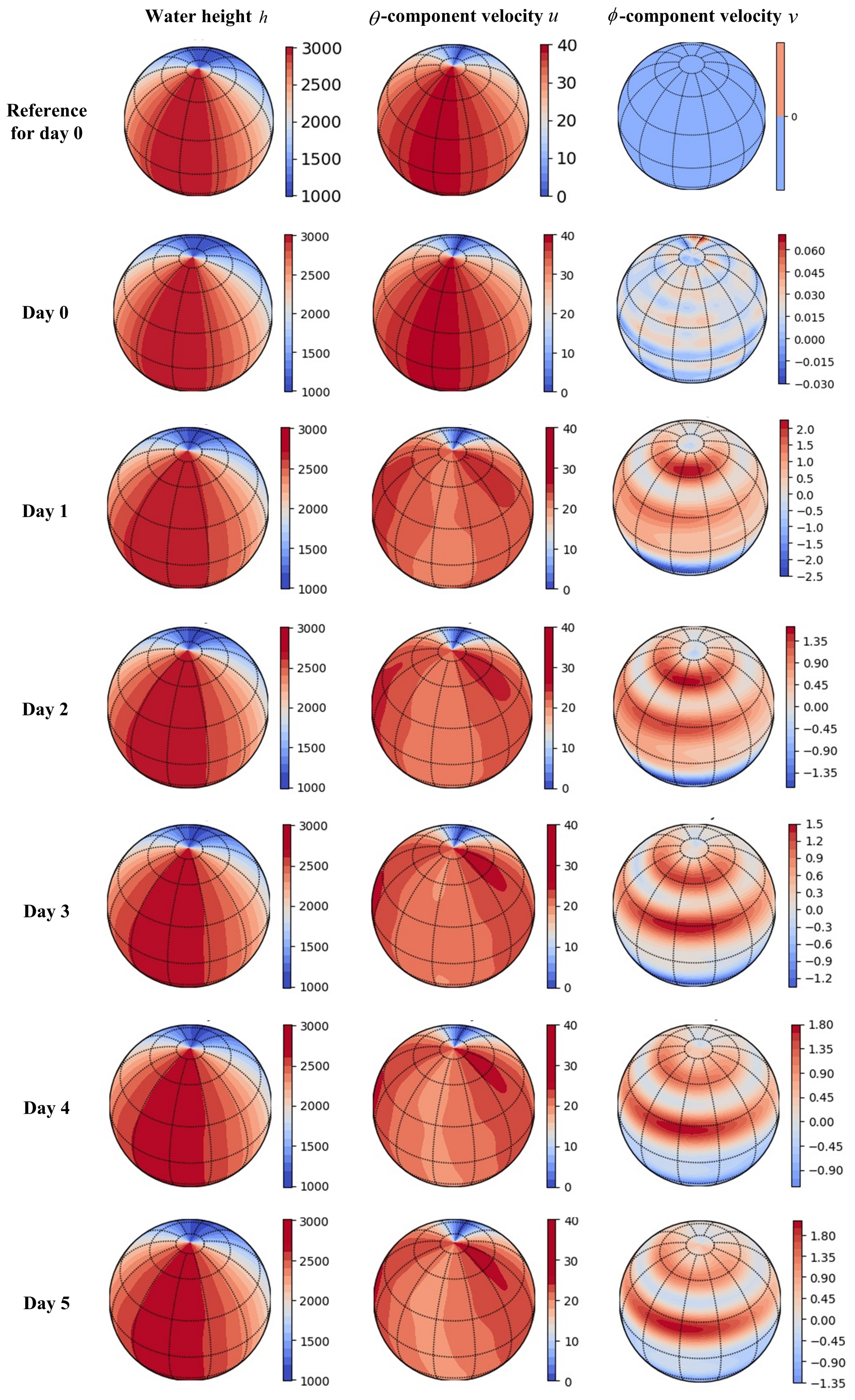


Figure 7: Results of GeoPINS with FD for 2-D Shallow water equation on the sphere in latitudelongitude coordinates. The first row is the exact solution for day 0 (reference). The second to seventh rows are solutions of GeoPINS with FD from day 0 to day 5.

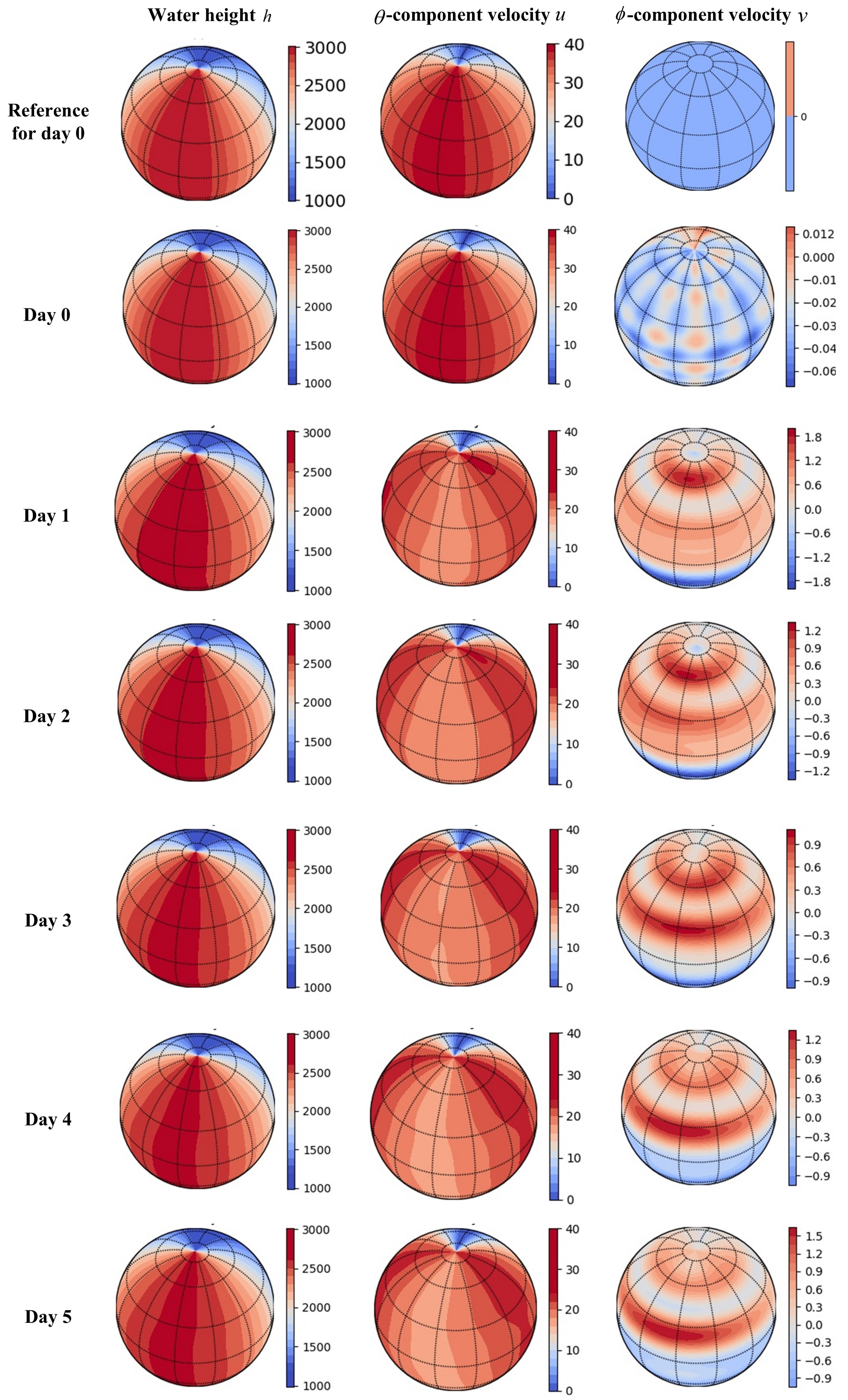


Figure 8: Results of GeoPINS with FFT PS for 2-D Shallow water equation on on the sphere in latitudelongitude coordinates. The first row is the exact solution for day 0 (reference). The second to seventh rows are solutions of GeoPINS with FFT PS from day 0 to day 5.

H are characteristic scales of large-scale atmospheric phenomena. Specifically,

$$L = a[\mathrm{m}] = 6371220\,\mathrm{m}, \quad T = 1[\mathrm{day}] = 86400[\mathrm{s}], \quad U = L/T\left[\mathrm{m}\cdot\mathrm{s}^{-1}\right], \quad H = c\frac{L^2}{T^2 g}[\mathrm{m}]. \tag{24}$$

Global geostrophic flow is the only test case for which an exact solution is known. In order to fully verify the validity of GeoPINS in dynamic geometry-adaptive domains, we follow Williamson et al. (1992), the initial condition for this test case is,

$$\begin{aligned} u &= u_0\cos\phi, \quad v = 0 \\ gh &= gh_0 - \left(a\omega u_0 + \frac{u_0^2}{2}\right)\sin^2\phi, \end{aligned} \tag{25}$$

where $u_0 = 2\pi a/12$ m$\cdot$ days $^{-1}$ and $gh_0 = 2.94\cdot 10^4\ \mathrm{m}^2\cdot\mathrm{s}^{-2}$. The Coriolis parameter $f = 2\omega\sin\phi$. The angular velocity of Earths rotation $\omega = \frac{2\pi}{86400}\left[\mathrm{s}^{-1}\right]$. This test is to be run for 5 days. Thus, we consider dynamic spherical coordinates $\Omega_g = [t, \hat{\theta}, \hat{\phi}] = [0,5]\times[-\pi,\pi]\times[-\pi/2,\pi/2]$ as the physical domain and Cartesian coordinates $\Omega_r = [t,x,y] = [0,5]\times[0,500]\times[0,500]$ as the computational domain. $dt \times dx \times dy = 1 \times 2\pi/500 \times \pi/500$. The GeoPC loss can be obtained by combing Eq. 23 and Eq. 25,

$$\mathcal{L}_{\text{GeoPC}} = \min_{\theta} \|\mathcal{L}_{\text{Initial}}\|_{\Omega_g} + \lambda \|\mathcal{L}_{\text{non-dimensional shallow water equation}}\|_{\Omega_g}. \tag{26}$$

The boundary conditions on the sphere is enforced as hard constraint by a coordinate-transform layer Bihlo & Popovych (2022). In terms of GeoPINS with FD and GeoPINS with FFT PS, batch size = 1, total epochs = 15000. We use Adam optimizer with the learning rate 0.001 that decays by half every 50 epochs. The Fourier modes $k = \{4,4,4,4\}$ in time dimension due to the limitation of the tensor size in time dimension. The input channel is set to 3, including non-dimensional latitude coordinate $\hat{\theta}$, non-dimensional longitude coordinate $\hat{\phi}$, and non-dimensional time t. It is notice that predictions ($\hat{u}$, $\hat{v}$, $\hat{h}^*$) of GeoPINS need to be restored to the physical scale.

Visualization results of GeoPINS with FD and GeoPINS with FFT PS are shown in Fig. 7 and Fig. 8, respectively. According to comparisons between GeoPINS prediction results and reference results on day 0 in Fig. 7 and Fig. 8, GeoPINS can accurately predict the velocity and water height. It is also demonstrated that GeoPINS is an effective and stable dynamic geometry-adaptive solver for global dynamics simulation based on the 5-day simulation results. To quantitatively evaluate the validity of GeoPINS, we follow Bihlo & Popovych (2022); Williamson et al. (1992) in the choice of relative error measures at time t ($\mathrm{RE}_2(t)$), which are the L_2-error being weighted by the cosine of latitudes. Formulated as,

$$\mathrm{RE}_2(t) = \frac{\left(\sum_{i=1}^{N}\left(\hat{h}_m\left(t,\theta^i,\phi^i\right) - h\left(t,\theta^i,\theta^i\right)\right)^2\cos\phi^i\right)^{1/2}}{\left(\sum_{i=1}^{N} h\left(t,\theta^i,\phi^i\right)^2\cos\phi^i\right)^{1/2}}, \tag{27}$$

where $N = N_{lon}\times N_{lat} = 500\times 500$ denotes the total number of grid points being used for the error. $\hat{h}_m$ and h represent prediction results of GeoPINS and exact solutions of the nonlinear zonal geostrophic flow test case Williamson et al. (1992), respectively. The relative errors of GeoPINS with FD and GeoPINS with FFT PS on the final time t = 5 days are presented in Table 3. The results confirm that GeoPINS is an accurate dynamic solver for global dynamics simulation. In addition, compared with GeoPINS with FD, GeoPINS with FFT PS is generally better performing for solving shallow water equation on the sphere.