Quantum Physics Informed Neural Network for the prediction of hurricanes

Gilberto Juarez Rangez, Fernando Torres Leal, Maria Jose Díaz Sánchez, Daniela Rodríguez Trujillo.
(Dated: August 10, 2024)

Abstract: Effective hurricane prediction is essential for minimizing the devastation caused by these powerful storms, yet traditional forecasting methods, reliant on complex models and extensive historical data, often involve significant computational costs. Quantum machine learning (QML) presents a novel approach by harnessing the computational prowess of quantum computing alongside advanced machine learning techniques, potentially enhancing prediction accuracy and improving early warning systems. This study explores the application of Variational Quantum Machine Learning (VQML) for hurricane prediction, specifically utilizing quantum circuits in place of classical neural networks. Our experiments utilized N = 4 qubits and l = 4 ansatz layers. Results indicate that while current hardware limitations restrict the number of training epochs, this work highlights the preliminary functionality of the algorithm and its prospective advantages on a fully operational quantum computer.

I. INTRODUCTION

Hurricane prediction is a critical component of disaster preparedness and risk management, aiming to enhance our ability to anticipate and respond to these types of weather phenomena. Hurricanes, whose wind speeds exceeds 74 mph, are cyclonic storms characterized by organized thunderstorm activity [1]. Their development is closely linked to the energy they gain from the warm tropical sea [1, 2], they typically take up to half a day to evolve or change their path, and their lifespan ranges from a few days to several weeks [2].

Hurricanes are a leading cause of economic damage both in the continental United States and globally [3]. Among all recorded weather disasters in U.S. history, hurricanes have caused the most deaths and destruction [4, 5]. As a result, there has been a growing emphasis on improving hurricane monitoring and forecasting.

Hurricane forecast models began to develop in the 1950s with the advent of aircraft reconnaissance, which provided accurate data on a hurricane's position and intensity [2, 6]. These early models were divided into dynamic and statistical types. Dynamic models, utilizing supercomputers, solve equations based on atmospheric physics and motion, while statistical models use historical hurricane data to predict future behavior [6]. By the 1970s, advancements led to the creation of statistical-dynamical models, combining the strengths of both approaches [5, 7].

In the 1990s and 2000s, the National Hurricane Center (NHC) employed prominent statistical-dynamical models like SHIPS and GFDL[8]. SHIPS used various predictors to forecast intensity, while GFDL aimed to dynamically forecast both intensity and structure. The subsequent introduction of models during 2007-2019 such as LGEM and HWRF, with HWRF outperforming GFDL in accuracy, highlighted ongoing improvements[8].

Despite advances, dynamic models still face challenges, especially with rapid intensification (RI), which has shown a slow improvement in the last decades with most of the models currently employed in the NHC[6, 8].

Hurricane prognosis is vital for risk mitigation, it is an important area of weather prognosis that directly affects the general population. Hurricanes have devastating effects on the locations they afflict, as was evidenced on 2023 as hurricane Franklin appeared on the Caribbean Sea.

Hurricane Franklin occurred from August 20th to September 1st, 2023. It was officially classified as a hurricane early on August 20, when it formed southwest of the Caribbean Sea, initially classified as a tropical storm [9]. Over the following days, it intensified into a hurricane until August 29 when Franklin reached its peak intensity of 149.601 mph impacting the Dominican Republic where it caused heavy floods and landslides with a rainfall between 244.6mm-300mm [9, 10] and left more than 1 million residents without water.Overall, media reports estimate that the storm caused approximately "$90 million in damage across the country, primarily due to freshwater flooding that affected homes and infrastructure."[9].

Hurricanes such as Franklin make it imperative to innovate in methods that can accurately predict hurricane behavior without the great computational cost that some traditional models have. In this work we focus on using Physics Informed Neural Networks (PINNs) along with algorthms that utilize Quantum Machine Learning (QML) [11]. PINNs are neural networks that are trained to solve task while maintaining the restrictions of any given laws of physics described by general nonlinear partial differential equations [12].PINNs have been used to simulate tropical hurricanes' wind and pressure fields using real life observations [13].

On the other hand, QML is the intersection between artificial intelligence and quantum technology [14]. However, due to the high number of quantum operations and qubits required, QML algorithms require quantum computing which provides a challenge on its own [15]. The solutions found and the one used for this work are Variational QML algorithms in which the classical neural network is replaced with a quantum circuit [11]. This circuits provide a unable output that can be used in a loss function and they can also be made to comply with physical laws. This type of approach has been used previously for solving the barotropic vorticity equation, related to

atmospheric movements, proving the adaptability of this method to recreate and predict weather phenomena [11].

II. METHODS

A. Data

In order to study the evolutionary behavior of Franklin, we used the data from the Climate Data Store from Copernicus Climate Change Service (C3S) at ECMWF [16]. The data was recovered by ERA5 which provides "hourly estimates of a large number of atmospheric, land and oceanic climate variables. The data covers the Earth on a 31km grid and resolve the atmosphere using 137 levels from the surface up to a height of 80km" [17, 18].

For the equations to work with, the data collected included the wind velocity (m/s) components in orthonormal 2D format: u and v. Here, u represents the eastward component of the wind, and v represents the northward component. These components are used to determine the horizontal wind speed and direction [17].

Longitude (ϕ), Latitude (λ), and geopotential (Φ) were the other variables taken into account. The geopotential (m^2/s^2) reflects the gravitational potential energy per unit mass at a given location relative to mean sea level [17].

The data coverage we use is confined to the region of the hurricane's most active trajectory, ranging from 5°N to 40°N, and 75°°W to 53°W from on August 20th to August 31st. The only pressure level used is 850 hPa. We employ a temporal resolution of 3 hours, with data points starting at 00:00 until 21:00 of each day.

The raw data had 88 points of time, 89 point of longitude and 181 point of latitude for each variable. We noticed that the amount of training epochs available for our implementation is highly limited by time and hardware when contrasting with [13] where for a classical implementation of the prediction for a tropical cyclone, 120000 training epochs were used. Due to this, depending on the cases we trained the VQC a different number of epochs.For computational purposes only 61 point of latitude, 30 point of longitude and 10 points of time were taken.

The longitude and latitude data can be mapped into x and y values through the arc longitude using the radius of the Earth $R = 6371\,\text{km}$ as,

$$x = R\lambda, \tag{1}$$

$$y = R\phi. \tag{2}$$

The performance of the variational circuit along with the PINN is benefiated of an encoding of the inputs and outputs. We define the following linear transformations for the input variables,

$$\begin{aligned} t' &= \alpha_t t + \beta_t, \\ x' &= \alpha_x x + \beta_x, \\ y' &= \alpha_y y + \beta_y, \end{aligned} \tag{3}$$

where $t', x', y' \in [0, 1]$. A similar thing is defined for the output variables,

$$\begin{aligned} u' &= \alpha_u u + \beta_u, \\ v' &= \alpha_v v + \beta_v, \\ \Phi' &= \alpha_\phi \Phi + \beta_\phi, \end{aligned} \tag{4}$$

where $u', v', \Phi' \in [-1, 1]$.

B. Variational Quantum Circuit architecture and training

A variational quantum circuit is composed of a series of parameterized unitary operators [19]. This allows the circuit to act as a quantum version of a classical Neural Network [19]. For the code implementation we use Python with the Pennylane framework and PyTorch package.

These quantum circuits are usually composed of two parts of a quantum feature map (FM) and a set of ansatz layers. The FM is used to encode the classical inputs t', u', v' into the Hilbert space [11], there are several architectures available, in this case we use a serial quantum feature map $\hat{U}_{\text{FM}}(x', y', t')$, in this kind of FM, some intermediate parametric unitary gates are added between each feature to change the computational basis, otherwise there would be loss of information [11].

This implementation can be seen in Figure 1 in the enclosed dashed line where we use combination of y-axis rotations $\hat{R}_y(\zeta_{m,i}m)$, where m represent any of the inputs and i numbers the parameters. The $\hat{U}_{\text{BEL}}$ operator corresponds to a *BasicEntangledLayer* object from Pennylane [20] that applies a layer of x-axis rotations $\hat{R}_x(\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a vector of learnable parameters followed by a *ring* of $\hat{C}_{\text{NOT}}$ operators [11].

The ansatz layers which are expressed mathematically as $\hat{U}_A(\boldsymbol{\theta})$, which can be decomposed into a series of layers of the form,

$$\hat{U}_A(\boldsymbol{\theta}) = \hat{U}_{A_1}(\boldsymbol{\theta}_1)\hat{U}_{A_2}(\boldsymbol{\theta}_2)\dots\ , \tag{5}$$

our implementation is a series of $\hat{U}_{\text{BEL}}$ operator as layers. Figure 1 shows an implementation with three ansatz layers. The resulting state of this operation can be represented as,

$$|\psi\rangle = \hat{U}_A(\boldsymbol{\theta})\hat{U}_{\text{FM}}(x', y', t')|0\rangle. \tag{6}$$

In order to retrieve the classical information this can be done by the expectation value $\langle\psi|\hat{C}|\psi\rangle$, where $\hat{C}$ is a cost operator [11] defined as,

$$\hat{C} = \sum_{n=1}^{N} \hat{Z}^n, \tag{7}$$

where N is the number of qubits available, and $\hat{Z}$ is the usual Pauli $\hat{Z}$ operator.

For training, we define the vector of inputs $\boldsymbol{x}_i = (t'_i, x'_i, y'_i)$, and outputs $\boldsymbol{y}_i = (u'_i, v'_i, \Phi'_i)$. The output prediction of the VQC is represented by,

$$\tilde{\boldsymbol{y}}_i = \langle \psi(\boldsymbol{x}_i) | \hat{C} | \psi(\boldsymbol{x}_i) \rangle, \tag{8}$$

with $\tilde{\boldsymbol{y}}_i = (u_{\mathrm{VQC}}, v_{\mathrm{VQC}}, \Phi_{\mathrm{VQC}})$ being the predictions of the quantum circuit. Since this expectation value ranges $[-N, N]$, we need to introduce a linear learnable linear transformation such that the VQC prediction takes the form [11],

$$\tilde{\boldsymbol{y}}'_i = \alpha_q \tilde{\boldsymbol{y}}_i + \beta_q, \tag{9}$$

where $\tilde{\boldsymbol{y}}'_i$ is the transformed prediction of the outputs by the VQC, and α_q, β_q are learnable parameters to adjust the value of the prediction to a physical range. Moreover, we trained it by splitting the data on a fifty-fifty proportion [21]; from our ten time data, five went into training and the other five into testing.

Using hybrid quantum computing using Pennylane and Pytorch we use the classical LBFGS optimizer to adjust the parameters and approximate $\tilde{\boldsymbol{y}}'_i \approx \boldsymbol{y}_i$ as it has shown good result for PINN implementations []. We use a loss function as a reference for optimizing the parameters, taking into account both data and physics. This is shown in depth in Section II D. It is also worth mentioning that in practice, with a quantum computer if the cost operator is more complicated a Hadamard test can be used to compute the expectation values [11].

C. Physical equations to constrain PINN

The model including the equations and constraints that will be explained in this section heavily relies in the method described in [13]. To model the behavior of tropical cyclones and hurricanes it quickly becomes a problem of geophysical flow. An useful equation to model the dynamics of wind is the Navier-Stokes momentum equation in a rotational reference frame for incompressible flow [2, 13],

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} + 2\boldsymbol{\Omega} \times \boldsymbol{v} = -\frac{1}{\rho} \nabla p + \nabla \cdot \boldsymbol{\tau}, \tag{10}$$

where the wind velocity vector field is $\boldsymbol{v} = u\hat{\boldsymbol{x}} + v\hat{\boldsymbol{y}} + w\hat{\boldsymbol{z}}$; $\boldsymbol{\Omega} = (f/2)\hat{\boldsymbol{z}}$ is the Earth's angular velocity vector with rotation rate $\Omega = 7.2921 \times 10^{-5}\,\mathrm{s}^{-1}$, Coriolis parameter $f = 2\Omega \sin\phi$, and ϕ being the latitude; ρ is the density field, p is the pressure field, and $\boldsymbol{\tau}$ is the deviatoric stress tensor.

On the other hand, the conservation of mass is needed and it is expressed as the following continuity equation [13, 22],

$$\frac{\partial \rho}{\partial t} + \nabla(\rho \boldsymbol{v}) = 0. \tag{11}$$

Making a series of assumptions we can yield to a manageable set of equations which correctly guides the model without being computationally expensive [13]. Since the phenomenon is taking place at high altitude we consider the friction term to be negligible [13]. Since the vertical velocity measurements are not reliable, instead the pressure is used as the vertical coordinate so $w = \mathrm{d}p/\mathrm{d}t$ [11, 13].

We consider the hydrostatic approximation for simplification sake, it takes the form [23],

$$\frac{\partial p}{\partial z} = -\rho g, \tag{12}$$

where g is the gravitational acceleration. A geopotential $\Phi = gz$ is defined in order to relate it to the hydrostatic pressure flow since it is and observable parameter. Now, for the rotational frame of reference we take the beta-plane approximation to express the Coriolis parameter as [24],

$$f = f_0 + \beta y, \tag{13}$$

where f_0 is the Coriolis parameter at latitude ϕ_0, $\beta = (\partial f/\partial y)|_{\phi_0}$ is the Rossby parameter [24], and y is the distance from ϕ_0 in the latitude direction. We are only interested in the horizontal components since those measurements are easily obtained. These approximations allows us to express the Navier-Stokes equations (Eq. (14), and Eq. (15)) and the continuity equation (Eq. (16)) separated in by horizontal components as,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial p} - (f_0 + \beta y)v = -\frac{\partial \Phi}{\partial x}, \tag{14}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial p} + (f_0 + \beta y)u = -\frac{\partial \Phi}{\partial y}, \tag{15}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial p} = 0. \tag{16}$$

The derivation of these equations is available in Appendix A and Appendix B. The introduction of the linear transformation into the input and output variables from Eq. (3) and Eq. (4) changes the form of the equations. The equations that were used for the code implementation are derived and shown in Appendix C.

D. Loss function

The loss function $\mathcal{L}$ is introduced in order to train the variational quantum circuit. It is composed of a part associated to the data ($\mathcal{L}_{\mathrm{Data}}$) and the other to the physical equations ($\mathcal{L}_{\mathrm{Physics}}$) such that [13],

$$\mathcal{L} = (1 - \gamma)\mathcal{L}_{\mathrm{Data}} + \gamma \mathcal{L}_{\mathrm{Physics}}, \tag{17}$$

where $\gamma \in [0, 1]$ is a parameter to control the relevance of each loss. As aforementioned, the both input and output parameters were linearly transformed, so in this case the losses take the form [13],

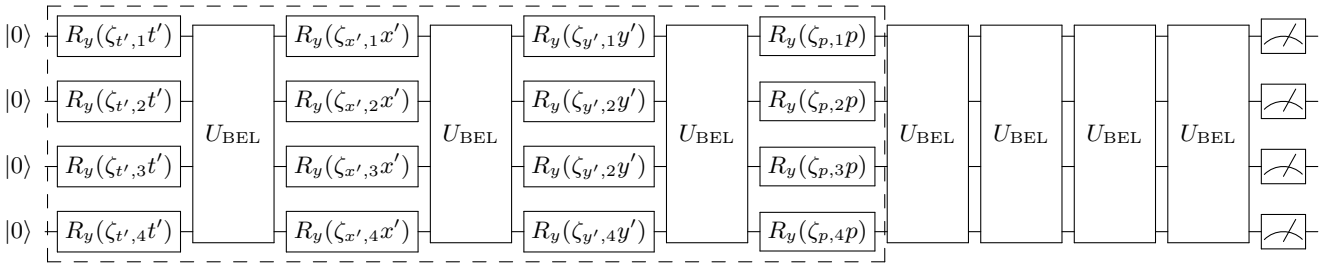

FIG. 1. Circuit diagram demonstrating the quantum feature map (enclosed in dashed line). Each feature t', x', y' is encoded sequentially in a serial quantum feature map with trainable ansatz between layers. The pressure p is present in the rotations and it is a constant as the data was taken at 850 hPa. After the feature map, a series of 4 ansatz layers is added to end with measuremet. R_y represents rotations in the y-axis, an U_{BEL} represents a layer of R_x rotations in x-axis followed by a ring of CNOT operators known as a basic entangled layer by Pennylane.

$$\mathcal{L}_{\mathrm{Data}} = \frac{1}{n}\sum \sqrt{u'^2+v'^2}\Big[(u_{\mathrm{VQC}}-u')^2+\dots (v_{\mathrm{VQC}}-v')^2+(\Phi_{\mathrm{VQC}}-\Phi')^2\Big], \tag{18}$$

$$\mathcal{L}_{\mathrm{Physics}} = \frac{1}{m}\sum\Bigg[\left(\frac{\partial u'}{\partial t'}+u'\frac{\partial u'}{\partial x'}+v'\frac{\partial u'}{\partial y'}+\dots +w\frac{\partial u'}{\partial p'}-(f_0+\beta y')v'+\frac{\partial \Phi'}{\partial x'}\right)^2+\dots +\left(\frac{\partial v'}{\partial t'}+u'\frac{\partial v'}{\partial x'}+v'\frac{\partial v'}{\partial y'}+w\frac{\partial v'}{\partial p}+\dots +(f_0+\beta y')u'+\frac{\partial \Phi'}{\partial y'}\right)^2+\left(\frac{\partial u'}{\partial x'}+\frac{\partial v'}{\partial y'}+\frac{\partial w}{\partial p}\right)^2\Bigg], \tag{19}$$

where u_{VQC}, v_{VQC}, and Φ_{VQC} are the outputs prediced by the variational quantum circuit. Transformed variables are used on Eq. (18) and Eq. (19), in the code, this translates to using the equations in Appendix C. The data loss on Eq. (18) calculates a mean square error (MSE) of the predicted data and the train data. The physical loss in Eq. (19) is calculated as the mean of the numerical evaluation using a Parameter-shift rule to evaluate the partial differential equations.

III. RESULTS AND DISCUSSION

In this section we present the results of the supervised learning by the VQC. For each experiment we used $N = 4$ qubits, and $l = 4$ ansatz layers. With these values $\dim(\boldsymbol{\theta}) = 64$, so there are 64 trainable parameters. The implementation was run using Google Colab with a T4 and A100 graphics card.

The results presented have the only purpose of showing the functionality of the algorithm and stating the advantages if run in a real quantum computer.

A. Data-based training

To test that the VQC works, we start with a parameter $\gamma = 0$, meaning that only the loss from $\mathrm{L}_{\mathrm{Data}}$ is going to have a role in the optimization of parameters. The model was trained using a LBFGS optimizer for 200 epochs. Figure 2 depicts the predictions of the Neural network, showing that it learned the patterns of the training data showing that the patterns of intensity match, specially, Figure 2e and Figure 2f which can be caused by the smoothness of the geopotential function and its consistency throughout time which doesn't happen for the wind velocity components.

For the training set, it is not capable of reproduce any pattern due to the lack of information from the Navier-Stokes equation and the continuity equation. This is noticeable from Figure 3 where the convergence of the loss function show clear overfitting.

B. Training on data and physics

Now, we explore the learning of the VQC by incorporating values $\gamma = 0.05$ and $\gamma = 0.5$. We don't include experiments for high values of γ because of the high computational cost.

1. *Parameter* $\gamma = 0.05$

Starting with a low value of γ, the model was trained for a 100 epoch. In this case we trade off the training with the data set for physical information. This is reflected clearly in Figure 5 where the convergence of the loss function is much closer than the case with $\gamma = 0$. On the other hand, Figure 4 show that the wind velocity component predicts worse the intensity patterns, specially in the zones of high intensity, this can be due to the lack of data training but also to the hydrostatic approximation which does not take into account the ver-

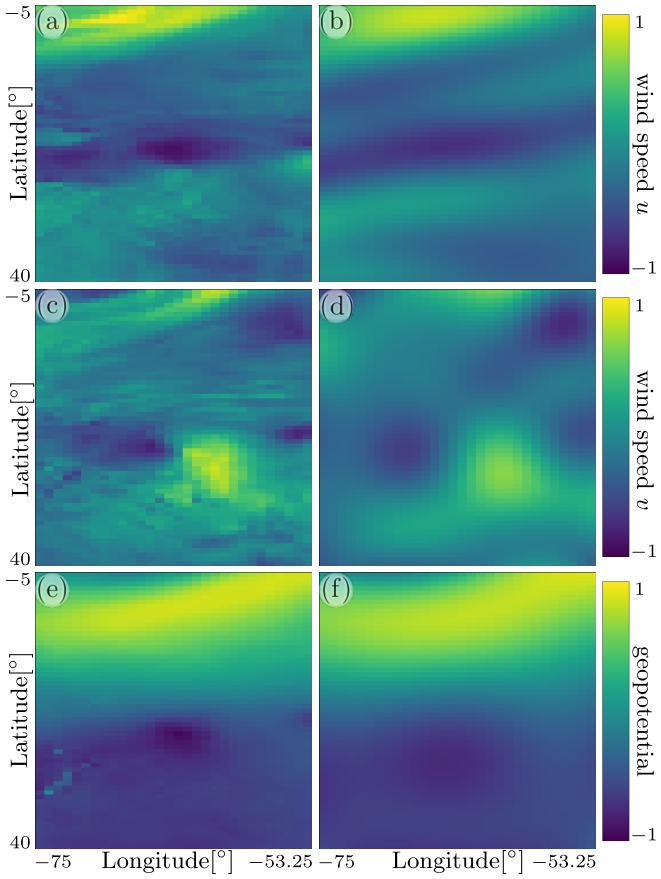

FIG. 2. Prediction of output variables wind velocity component u and v, geopotential Φ ((b), (d), (f)) and comparison with original data ((a), (c), (e)) using $\gamma = 0$.

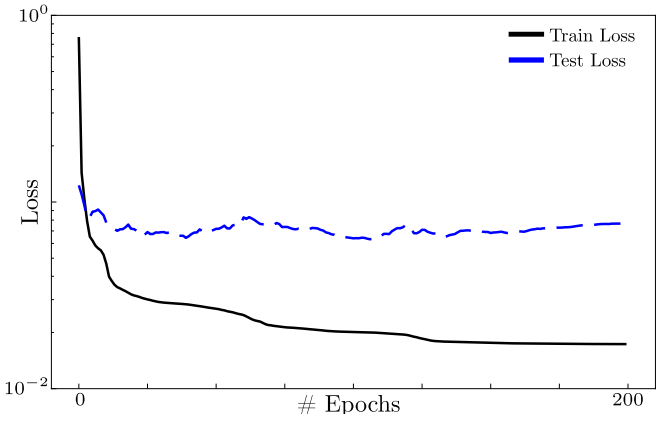

FIG. 3. Convergence of loss function with train data and test data for $\gamma = 0$.

tical wind velocity [13]. The geopotential prediction still remains the closest out of the three output variables.

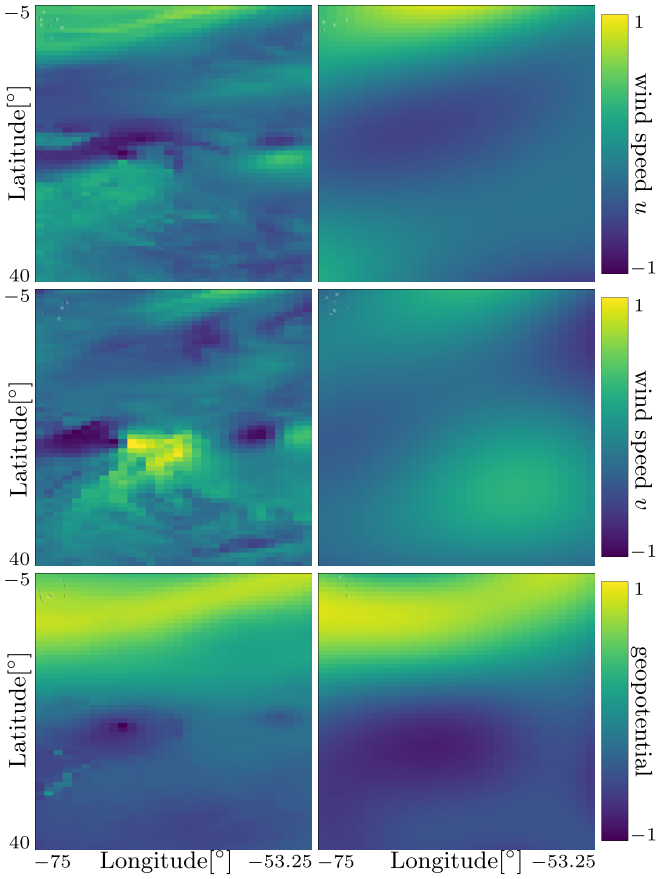

FIG. 4. Prediction of output variables wind velocity component u and v, geopotential Φ ((b), (d), (f)) and comparison with original data ((a), (c), (e)) using $\gamma = 0.05$.

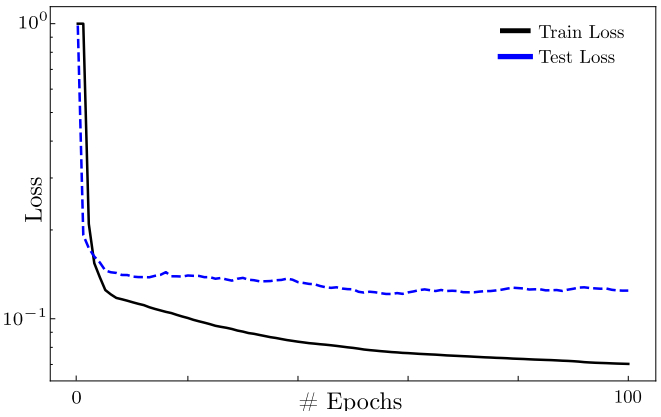

FIG. 5. Convergence of loss function with train data and test data for $\gamma = 0.05$.

2. *Parameter* $\gamma = 0.5$

For the case of $\gamma = 0.5$ the contributions of the loss function for both data and the physics is the same. The model is trained for 40 epochs. The results of the convergence of the loss functions show a reduction of total

loss for both sets of data as shown in Figure 7. Nevertheless, there is no consistency in the data which reflects bad for predictions. This can be seen on Figure 6 where is noticeable that the predicted values tend to a constant.

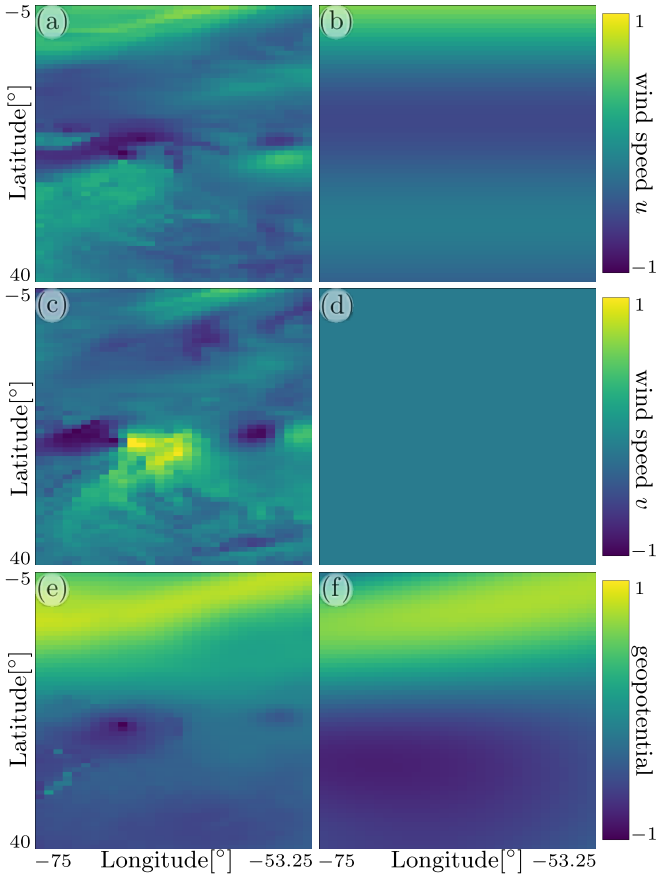

FIG. 6. Prediction of output variables wind velocity component u and v, geopotential Φ ((b), (d), (f)) and comparison with original data ((a), (c), (e)) using $\gamma = 0.5$.

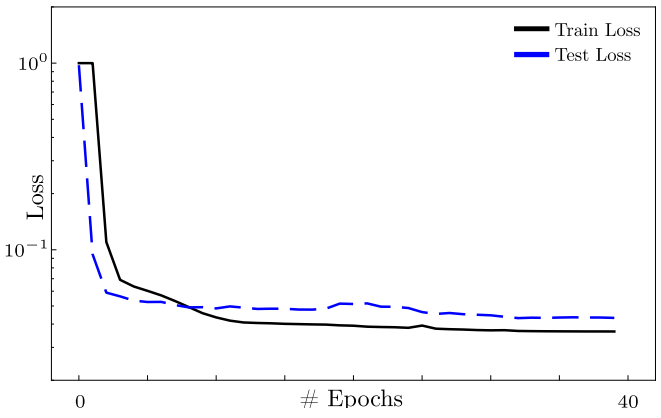

FIG. 7. Convergence of loss function with train data and test data for $\gamma = 0.5$.

3. *Absolute error*

In order to test where the model fails the most, we define and absolute error as $\delta = |\tilde{\boldsymbol{y}} - y'|$ and in Figure 8 we can see that the error is low but there's discrepancy of the intensity patterns, also, the highest error occurs in the zone of high wind velocity for component v which is consistent with the effects of the hydrostatic approximation aforementioned.

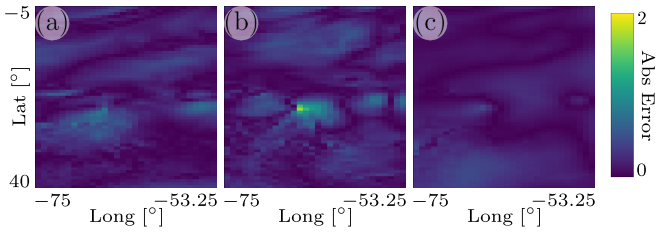

FIG. 8. Absolute error of the predictions for wind velocity component u and v, geopotential Φ ((a), (b), (c)) for $\gamma = 0.05$.

IV. CONCLUSION

In conclusion, the use of PINNs with quantum machine learning is a prospective candidate to enhance the methods of hurricane prediction. This results in this work shows that the model is capable of learning patterns accordingly and to use the differential equations to learn, but it is necessary a proper training that involves much more training epochs to get close to a useful prediction of the phenomenon. This is evident when contrasting with works like [13] where 120000 epochs where necessary to train a classical neural network for the same purpose.

A possible improvement for this model is to include an extra loss term with the standard deviation in order to avoid that constant predictions. Considering that this model with a small amount of training was capable of decently predict the intensity patterns, we infer that with more qubits and more complex ansatz we would see great results.

V. DATA AVAILABILITY

ERA5 hourly data on pressure levels is available in `https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-pressure-levels`

VI. CODE AVAILABILITY

The code implementing the Variational Quantum Circuit with PINN is availble in `https://github.com/ComfyBear41/Quantum-AI-for-Climate`.

[1] NOAA, Hurricanes (2020).
[2] H. E. Willoughby, E. Rappaport, and F. Marks, Natural Hazards Review , 45 (2007).
[3] P. J. Klotzbach, S. G. Bowen, R. Pielke, and M. Bell, Bulletin of the American Meteorological Society , 1359 (2018).
[4] NOAA, Hurricane costs (2023).
[5] R. C. Sheets, Weather and Forecasting **5**, 185 (1990).
[6] NSF, Hurricanes: Science and Society: Brief History of Hurricane Forecast Models (2020).
[7] IEE, Hurricane tracking technology: Advancements and opportunities (2024).
[8] NSF, Hurricanes: Science and Society: National Hurricane Center Forecast Process (2020).
[9] J. L. B. II, Hurricane franklin (2024).
[10] ONAMET, Onamet informa tormenta franklin producirá aguaceros y oleaje anormal rd (2023).
[11] B. Jaderberg, A. A. Gentile, A. Ghosh, V. E. Elfving, C. Jones, D. Vodola, J. Manobianco, and H. Weiss, Potential of quantum scientific machine learning applied to weather modelling (2024), arXiv:2404.08737 [quant-ph].
[12] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Journal of Computational Physics **378**, 686 (2019).
[13] R. Eusebi, G. A. Vecchi, C. Y. Lai, and M. Tong, Communications Earth and Environment **5**, 1 (2024).
[14] A. Melnikov, M. Kordzanganeh, A. Alodjants, and R. K. Lee, Quantum machine learning: from physics to software engineering (2023), arXiv:2301.01851.
[15] F. Tennie and T. N. Palmer, Bulletin of the American Meteorological Society , E488 (2023).
[16] Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., Thépaut, J-N., Era5 hourly data on pressure levels from 1940 to present. (2023).
[17] ECMFW, Ecmwf reanalysis v5 (era5) (2019).
[18] ECMFW, Era5: data documentation (2024).
[19] B. Ogur and I. Yilmaz, Quantum Inf. Comput. , 181 (2023).
[20] V. Bergholm, J. Izaac, M. Schuld, C. Gogolin, S. Ahmed, V. Ajith, M. S. Alam, G. Alonso-Linaje, B. Akash-Narayanan, A. Asadi, J. M. Arrazola, U. Azad, S. Banning, C. Blank, T. R. Bromley, B. A. Cordier, J. Ceroni, A. Delgado, O. D. Matteo, A. Dusko, T. Garg, D. Guala, A. Hayes, R. Hill, A. Ijaz, T. Isacsson, D. Ittah, S. Jahangiri, P. Jain, E. Jiang, A. Khandelwal, K. Kottmann, R. A. Lang, C. Lee, T. Loke, A. Lowe, K. McKiernan, J. J. Meyer, J. A. Montañez-Barrera, R. Moyard, Z. Niu, L. J. O'Riordan, S. Oud, A. Panigrahi, C.-Y. Park, D. Polatajko, N. Quesada, C. Roberts, N. Sá, I. Schoch, B. Shi, S. Shu, S. Sim, A. Singh, I. Strandberg, J. Soni, A. Száva, S. Thabet, R. A. Vargas-Hernández, T. Vincent, N. Vitucci, M. Weber, D. Wierichs, R. Wiersema, M. Willmann, V. Wong, S. Zhang, and N. Killoran, Pennylane: Automatic differentiation of hybrid quantum-classical computations (2022), arXiv:1811.04968 [quant-ph].
[21] R. Medar, V. S. Rajpurohit, and B. Rashmi (2017).
[22] L. Davidson, in *Advanced Approaches in Turbulence*, edited by P. Durbin (Elsevier, 2021) pp. 161–241.
[23] NOAA, The hydrostatic equation.
[24] C. Rossby, H. Willett, M. Holmboe, J. Namias, L. Page, R. Allen, *et al.*, Journal of Marine Research (1939).

APPENDIX

Appendix A: Derivation of Navier-Stokes equations with approximations

We start with the Navier-Stokes equation with a rotational frame of reference,

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} + 2\boldsymbol{\Omega}\times\boldsymbol{v} = -\frac{1}{\rho}\nabla p + \nabla\cdot\boldsymbol{\tau}, \tag{A1}$$

where $\boldsymbol{v} = u\hat{\boldsymbol{x}} + v\hat{\boldsymbol{y}} + w\hat{\boldsymbol{z}}$ is the wind velocity vector field, $\boldsymbol{\Omega} = (f/2)\hat{\boldsymbol{z}}$ is the earth's angular velocity vector and f the Coriolis parameter, ρ is the density field, p is the pressure field, and $\boldsymbol{\tau}$ is the deviatoric stress tensor. We will also define a geopotential $\Phi = gz$ which will be needed.

We can identify the first to terms of the LHS of the equation as the material derivative of the wind velocity, the Coriolis force term $2\boldsymbol{\Omega}\times\boldsymbol{v}$, the hydrostatic pressure gradient $\frac{1}{\rho}\nabla p$, and $\nabla\cdot\boldsymbol{\tau}$ which is related to the friction.

If we consider high altitudes where the friction can be consider negligible, then we can drop the deviatoric tensor term and we get,

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v}\cdot\nabla\boldsymbol{v} + 2\boldsymbol{\Omega}\times\boldsymbol{v} = -\frac{1}{\rho}\nabla p. \tag{A2}$$

Then, the hydrostatic approximation takes the form,

$$\frac{\partial p}{\partial z} = -\rho g, \tag{A3}$$

using the chain rule, we can express the following,

$$\frac{\partial p}{\partial z} = \frac{\partial p}{\partial \Phi}\frac{\partial \Phi}{\partial z} = \frac{\partial p}{\partial \Phi}g, \tag{A4}$$

and then remembering the definition of the geopotential Φ we get the following differential equation,

$$\frac{\partial p}{\partial \Phi} = \frac{1}{g}\frac{\partial p}{\partial z} = -\rho, \tag{A5}$$

integrating we get to,

$$p = -\int \rho \, \mathrm{d}\Phi = -\rho\Phi + p_0. \tag{A6}$$

If we replace the above definition of p into the hydrostatic pressure term we get that,

$$-\frac{1}{\rho}\nabla p = -\nabla\Phi. \tag{A7}$$

Now, we consider the beta-plane approximation were the Coriolis parameter is expressed with a linear approximation,

$$f = f_0 + \beta y, \tag{A8}$$

where f_0 is the Coriolis parameter at latitude ϕ_0, $\beta = (\partial f/\partial y)|_{\phi_0}$ is the Rossby parameter, and y meridonial distance from ϕ_0. With this, the Coriolis term in the Navier-Stokes equations gets the form,

$$2\boldsymbol{\Omega} \times \boldsymbol{v} = (f_0 + \beta y)\ \hat{\boldsymbol{z}} \times \boldsymbol{v}. \tag{A9}$$

With all the approximations mentioned, the Navier-Stokes equation gets to the form,

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} + (f_0 + \beta y)\ \hat{\boldsymbol{z}} \times \boldsymbol{v} = \nabla \Phi. \tag{A10}$$

We also consider pressure to be vertical coordinate so $w = \mathrm{d}p/\mathrm{d}t$ and we won't be focusing on the vertical component of wind field, so we get to the equations,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial p} - (f_0 + \beta y)v = -\frac{\partial \Phi}{\partial x}, \tag{A11}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial p} + (f_0 + \beta y)u = -\frac{\partial \Phi}{\partial y}. \tag{A12}$$

Appendix B: Continuity equation for incompressible flows

The continuity equation for wind flow is given by,

$$\frac{\partial \rho}{\partial t} + \nabla(\rho \boldsymbol{v}) = 0, \tag{B1}$$

where ρ is the density of the flow, and $\boldsymbol{v}$ is the wind velocity vector field. For incompressible fluids the conditions states that $\nabla \cdot \boldsymbol{v} = 0$. And in this case the density ρ is considered constant. So the equation reduces to,

$$\nabla \cdot \boldsymbol{v} = 0, \tag{B2}$$

or expressed explicitly, and taking into account that the vertical component is considered as the pressure, this yields to,

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial p} = 0. \tag{B3}$$

Appendix C: Derivation of Navier-Stokes equations to build the PINN

We define the following linear transformations for the input variables,

$$\begin{aligned} t' &= \alpha_t t + \beta_t, \\ x' &= \alpha_x x + \beta_x, \\ y' &= \alpha_y y + \beta_y, \end{aligned} \tag{C1}$$

where $t', x', y' \in [0, 1]$. A similar thing is defined for the output variables,

$$\begin{aligned} u' &= \alpha_u u + \beta_u, \\ v' &= \alpha_v v + \beta_v, \\ \Phi' &= \alpha_\phi \Phi + \beta_\phi, \end{aligned} \tag{C2}$$

where $u', v', \Phi' \in [-1, 1]$. This change of variables, using the chain rule has the following effect on the Navier-Stokes equation and the continuity equation,

$$\frac{1}{\alpha_u}\left[\alpha_t \frac{\partial u'}{\partial t'} + \alpha_x \left(\frac{u' - \beta_u}{\alpha_u}\right)\frac{\partial u'}{\partial x'} + \alpha_y \left(\frac{v' - \beta_v}{\alpha_v}\right)\frac{\partial u'}{\partial y'} + \dots + w\frac{\partial u}{\partial p}\right] - \left[f_0 + \beta\left(\frac{y' - \beta_y}{\alpha_y}\right)\right]\left(\frac{v' - \beta_v}{\alpha_v}\right) + \frac{\alpha_x}{\alpha_\phi}\frac{\partial \Phi'}{\partial x} = 0 \tag{C3}$$

$$\frac{1}{\alpha_v}\left[\alpha_t \frac{\partial v'}{\partial t'} + \alpha_x \left(\frac{u' - \beta_u}{\alpha_u}\right)\frac{\partial v'}{\partial x'} + \alpha_y \left(\frac{v' - \beta_v}{\alpha_v}\right)\frac{\partial v'}{\partial y'} + \dots + w\frac{\partial v}{\partial p}\right] + \left[f_0 + \beta\left(\frac{y' - \beta_y}{\alpha_y}\right)\right]\left(\frac{u' - \beta_u}{\alpha_v}\right) + \frac{\alpha_x}{\alpha_\phi}\frac{\partial \Phi'}{\partial y} = 0 \tag{C4}$$

$$\frac{\alpha_x}{\alpha_u}\frac{\partial u'}{\partial x'} + \frac{\alpha_y}{\alpha_v}\frac{\partial v'}{\partial y'} + \frac{\partial w}{\partial p} = 0. \tag{C5}$$