MONTH 2 ACTM MILESTONE REPORT FROM THE UNIVERSITY OF MARYLAND (SUBCONTRACTORS TEXAS A&M AND POTOMAC LLC)

Develop framework for the hybrid models identifying the known and unknown parts along with mathematical approaches (Tasks I and II)

1 Slab Ocean Model

1.1 The Model Equation

A *slab ocean model* is a thermodynamical model of the *ocean mixed layer*, which is the nearly constant temperature top layer of the ocean. The dynamical variable of the model is the *sea surface temperature (SST)*, which is the $T_{mix}(\lambda, \varphi, t)$ temperature of the ocean mixed layer. The SST depends on the geographical longitude λ, geographical latitude φ, and time t, but it does not depend on the depth. The thermodynamical equation for the water column of the ocean mixed layer at location (λ, φ) is

$$\frac{\partial T_{mix}}{\partial t}(\lambda, \varphi, t) = \frac{1}{\rho c_o h_{mix}(\lambda, \varphi)} \left[Q_{atm}(\lambda, \varphi, t) + Q_{ocn}(\lambda, \varphi, t) \right]. \tag{1}$$

In this equation, ρ_0=1026 kgm^{-3} is the density of ocean water in the mixed layer, c_0=3930 Jkg^{-1}K^{-1} is the specific heat of the ocean water, $h_{mix}(\lambda, \varphi)$ is the depth of the mixed layer, which depends on the location but not on time, $Q_{atm}(\lambda, \varphi, t)$ is the heat flux at the top of the mixed layer from the atmosphere, and

$$Q_{ocn}(\lambda, \varphi, t) = Q_{upw}(\lambda, \varphi, t) + Q_{adv}(\lambda, \varphi, t) \tag{2}$$

is the oceanic heat flux, which is the sum of the $Q_{upw}(\lambda, \varphi, t)$ heat flux at the bottom of the mix layer by upwellings and the heat flux $Q_{adv}(\lambda, \varphi, t)$ at the side walls of the water column by advection (see Fig. 1 for a schematic illustration). In short, $Q_{ocn}(\lambda, \varphi, t)$ represents the oceanic heat transport by the ocean circulation.

Equation (1) can be used as a prognostic equation for the SST in a numerical model, if the terms of the right-hand side are known and can be prescribed, or if they can be computed interactively in the model. The depth of the mixed layer $h_{mix}(\lambda, \varphi)$ is a prescribed empirical parameter, while the atmospheric heat flux $Q_{atm}(\lambda, \varphi, t)$ can be computed interactively, if the slab ocean model is coupled to an atmospheric model. While the oceanic heat flux $Q_{ocn}(\lambda, \varphi, t)$ has to be prescribed, there is no direct empirical knowledge about its value.

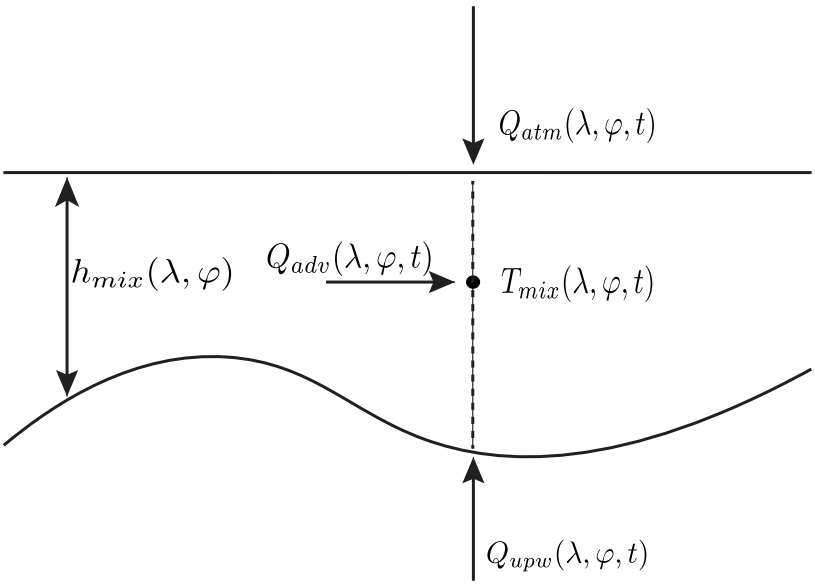

Figure 1: Schematic illustration of the slab ocean model.

1.2 Computation of the Atmospheric Heat Flux

An atmospheric model computes the heat flux from the ocean to the atmosphere. This heat flux is equal to $-Q_{atm}(\lambda,\varphi,t)$, because it describes the same transfer of heat, but from the point of view of the thermodynamics of the atmosphere rather than the ocean. Thus,

$$Q_{atm}(\lambda,\varphi,t) = Q_{sol}(\lambda,\varphi,t) - Q_{long}(\lambda,\varphi,t) - Q_{sen}(\lambda,\varphi,t) - Q_{lat}(\lambda,\varphi,t), \tag{3}$$

where $Q_{sol}(\lambda,\varphi,t)$ is the heating of the ocean mixed layer by solar radiation, $-Q_{long}(\lambda,\varphi,t)$ is its cooling by long wave radiation, $-Q_{sen}(\lambda,\varphi,t)$ is its cooling by the transfer of sensible heat from the ocean to the atmosphere in the atmospheric boundary layer, and $-Q_{lat}(\lambda,\varphi,t)$ is its cooling by the transfer of latent heat from the ocean to the atmosphere by evaporation. (In an atmospheric model, all these processes are represented by complicated parameterization schemes.) In a coupled simulation, all terms of the right-hand side of Eq. (3) can be computed interactively.

1.3 Estimation of the Oceanic Heat Flux

The climatological mean of Eq. (1) is

$$\overline{\frac{\partial T_{mix}}{\partial t}}(\lambda,\varphi) = \frac{1}{\rho c_o h_{mix}(\lambda,\varphi)} \left[\overline{Q}_{atm}(\lambda,\varphi) - \overline{Q}_{ocn}(\lambda,\varphi)\right], \tag{4}$$

where the bar denotes climatological mean. Assuming a stationary climate, the left-hand side is zero, which leads to the equation

$$\overline{Q}_{ocn}(\lambda,\varphi) = \overline{Q}_{atm}(\lambda,\varphi). \tag{5}$$

The SPEEDY implementation of the slab ocean model makes the additional assumption that

$$Q_{ocn}(\lambda,\varphi,t) = \overline{Q}_{ocn}(\lambda,\varphi). \tag{6}$$

Combining Eq. (5 and Eq. (6) leads to

$$Q_{ocn}(\lambda,\varphi,t) = \overline{Q}_{atm}(\lambda,\varphi). \tag{7}$$

Substituting $\overline{Q}_{atm}(\lambda,\varphi)$ for $Q_{ocn}(\lambda,\varphi,t)$ in Eq. (1) and making use of $\overline{\partial T_{mix}/\partial t} = 0$ yields

$$\frac{\partial T'_{mix}}{\partial t}(\lambda,\varphi,t) = \frac{Q'_{atm}(\lambda,\varphi,t)}{\rho c_o h_{mix}(\lambda,\varphi)}, \tag{8}$$

where $T'_{mix}(\lambda,\varphi,t) = T_{mix}(\lambda,\varphi,t) - \overline{T}_{mix}(\lambda,\varphi)$ and $Q'_{atm}(\lambda,\varphi,t) = Q_{atm}(\lambda,\varphi,t) - \overline{Q}_{atm}(\lambda,\varphi)$.

Using Eq. (8) in a coupled simulation requires the availability of an estimate of $\overline{Q}_{atm}(\lambda,\varphi)$. Such an estimate can be computed from a long (e.g., 30-year) simulation with the uncoupled atmospheric model, using observed values of the SST to compute the terms of the right-hand side of Eq. (3). Let $\overline{Q}^0_{atm}(\lambda,\varphi)$ be the estimate of $\overline{Q}_{atm}(\lambda,\varphi)$ obtained by this approach. When this estimate is used to compute $Q'_{atm}(\lambda,\varphi,t)$ in a coupled simulation, it is not guaranteed that $\overline{Q'_{atm}(\lambda,\varphi,t)} = 0$. Therefore, it is also not guaranteed that $\overline{\partial T_{mix}/\partial t} = 0$. In short, it is not guaranteed that the climatological mean of the SST in the coupled simulations is equal to the climatological mean of the observed SST that was used to force the uncoupled model simulation that produced $\overline{Q}^0_{atm}(\lambda,\varphi)$. To reduce the potential difference between the mean of the simulated SST and the mean of the observed SST, the equation of the slab ocean model of SPEEDY includes an extra term to relax the SST toward the observed climatology $\overline{T}_{mix}(\lambda,\varphi)$ of the SST. With this extra term, the dynamics of the SST in SPEEDY is described by

$$\frac{\partial T_{mix}}{\partial t}(\lambda,\varphi,t) = \frac{Q'_{atm}(\lambda,\varphi,t)}{\rho c_o h_{mix}(\lambda,\varphi)} - \tau_0^{-1} T'_{mix}(\lambda,\varphi,t) + \overline{T}_{mix}(\lambda,\varphi), \tag{9}$$

where the tunable model parameter τ_0 is a relaxation time constant.

2 Potential Approaches for the Application of ML

We hope to use ML to address the limitation of the slab ocean model that the oceanic heat transport by the ocean circulation cannot change in response to interactions with the atmosphere. Eliminating this limitation is important for the simulation of phenomena in which interactions between the atmospheric and oceanic circulation play an important role (e.g., El Nino). It is also essential for the modeling of climate change that cannot be done under the assumption of a stationary climate and an ocean circulation frozen in time.

2.1 ML-only Ocean Mixed Layer Model

This approach uses the parallel *reservoir computing (RC)* to replace Eq. (1) of the slab ocean model. The representation of the SST field and the global state vector for the ML model are the same as for SPEEDY. The input vector for the computation of a local state vector is defined by the usual strategy for overlaps and also includes the relevant grid-point values of $Q_{atm}(\lambda,\varphi,t)$. Including the latter components introduces the physics-based knowledge encapsulated in Eq. (1) into the ML model. Like the slab ocean model of SPEEDY, the ML model should use a time step of 1 day and 1-day averaged values of Q_{atm}. (Hopefully, we can reuse the code interface between the atmosphere and the ocean from SPEEDY.)

The attractive feature of this approach is that it does not require the estimation of $Q_{ocn}(\lambda,\varphi,t)$. Thus it does not require any assumption about the climate and the oceanic circulation. For the coupled atmosphere-system, it is a hybrid approach, because the atmospheric dynamics is modeled by a numerical model. The ML-only ocean affects the atmosphere through the regular parameterization schemes of the numerical model.

If the initial configuration does not perform as hoped, additional atmospheric variables can be added to the input of the reservoir computers. For instance, because we know that the wind drives the ocean surface currents that play a central role in the horizontal advection of heat in the ocean, we could also the horizontal wind components in the input vector.

2.2 Hybrid Ocean Mixed Layer Model

This approach applies CHyPP to the Ocean Mixed Layer. The physics-based component of the hybrid is Eq. (1). Initially, we could use the approach of SPEEDY for the estimation of Q_{ocn}. We could also develop improved approaches for the estimation of $Q_{ocn}(\lambda, \varphi, t)$ (e.g., approaches that reduce the discrepancy between the SST climate of the coupled model and the observed SST climate and/or introduce seasonal variability into $Q_{ocn}(\lambda, \varphi, t)$). In this approach, $Q_{atm}(\lambda, \varphi, t)$ does not have to be included in the input of the reservoir computer, because that information is already included in the physics-based component.

3 Mathematical Approaches for Improving Hybrid Model Stability

A significant challenge when training a machine learning model to predict time series data is ensuring the prediction's long-term stability. In our recent publications using our Combined Hybrid/Parallel prediction (CHyPP) method [1, 2], we were able to improve the long-term prediction stability by adding independent, mean-zero Gaussian noise vectors to the training data at each training time. Our CHyPP method was thus trained to both replicate the true system dynamics and to return to the true system trajectory after it had been perturbed away from it. One disadvantage of this method is that for any particular time during the training, our method is only trained to return from a particular perturbation corresponding to the noise vector that was added at that time. In the case where particular regions of phase space are only visited rarely during training (e.g., the system is high-dimensional or the duration of the training data is short), this creates the potential for instability due to the lack of damping of perturbations in all transverse directions within these regions.

We aim to further improve CHyPP's long-term prediction stability using the new technique of Linearized Multi-Noise Training (LMNT). This technique approximates the effect of adding many independent perturbations to the training data at each training time. In addition, it simplifies optimization of CHyPP's hyperparameters by converting the magnitude of the added noise into a regularization term, which can be optimized without re-computing any internal states. Prior to implementing this method on large climate systems, we will first test how well it is able to improve the stability of predictions of smaller model systems (e.g., the Kuramoto-Sivashinsky PDE with one spatial dimension). We now describe the training of a simple reservoir computer predictor using LMNT.

3.1 Training a Reservoir Computer using LMNT

3.1.1 Background

The open-loop (training) configuration of the simple reservoir computer is

$$r_n = f\left(r_{n-1}, u_{n-1}\right), \tag{10}$$

where n is the time index, r_n is a column vector representing the state of the reservoir computer at time index n, and u_n is a column vector representing the training data at time index n. Let $R = [r_1 \cdots r_N]$ be the matrix of reservoir states obtained during training and $U = [u_1 \cdots u_N]$ be the matrix of corresponding target outputs for prediction. Training a simple reservoir computer involves determining the matrix W such that $WR \approx U$. This is typically done using Tikhonov regularization, which minimizes

$$\sum_n \|Wr_n - u_n\|^2 + \|W\Gamma\|^2 = \|WR - U\|^2 + \|W\Gamma\|^2 \tag{11}$$

for some matrix Γ, usually chosen to be a multiple of the identity. Here, $\|A\|^2$ is the sum of the squares of the entries of a matrix A. The minimizer is

$$W = UR^T \left(RR^T + \Gamma\Gamma^T\right)^{-1}. \tag{12}$$

3.1.2 Multi-pass Training with Perturbations

Assume that we add K different sequences of perturbations $\varepsilon p_n^{(k)}$ to the time series of inputs u_n, where $k = 1, 2, \ldots, K$, and ε is a hyperparameter to be chosen during training. The resulting perturbed reservoir states after a single training pass are defined as

$$r_n^{(k)} = f\left(r_{n-1}^{(k)}, u_{n-1} + \varepsilon p_{n-1}^{(k)}\right). \tag{13}$$

We can then use regression on all of the pairs $(r_n^{(k)}, u_n)$ to find W such that $Wr_n^{(k)} \approx u_n$. Without Tikhonov regularization, this regression minimizes the cost function

$$J(x) = \frac{1}{K}\sum_{n,k}\left\|Wr_n^{(k)} - u_n\right\|^2. \tag{14}$$

Here, the $1/K$ normalization factor keeps the size of the expression roughly independent of K.

Let $\bar{r}_n$ be the mean over k of $r_n^{(k)}$, and let $q_n^{(k)} = r_n^{(k)} - \bar{r}_n$. Then

$$\begin{aligned} J(x) = \frac{1}{K}\sum_{n,k}\left\|Wr_n^{(k)} - u_n\right\|^2 &= \frac{1}{K}\sum_{n,k}\left\|W\bar{r}_n - u_n + Wq_n^{(k)}\right\|^2 \\ &= \frac{1}{K}\sum_{n,k}\left(\|W\bar{r}_n - u_n\|^2 + \left\|Wq_n^{(k)}\right\|^2 - 2\left(W\bar{r}_n - u_n\right)^T Wq_n^{(k)}\right) \\ &= \sum_n \|W\bar{r}_n - u_n\|^2 + \frac{1}{K}\sum_{n,k}\left\|Wq_n^{(k)}\right\|^2 \end{aligned} \tag{15}$$

since the mean over k of $q_n^{(k)}$ is zero. The first term is the usual sum-of-squared-residuals with r_n replaced by $\bar{r}_n$. The second term can be thought of as regularization that discourages W from having a large effect on the perturbations $q_n^{(k)}$ to the reservoir state. An additional Tikhonov regularization term can be added to $J(x)$.

3.1.3 Linearized Multi-Noise Training (LMNT)

If we assume that the use of r_n instead of $\bar{r}_n$ is unimportant for achieving long-term stability, then we can replace $\bar{r}_n$ with r_n and linearly approximate the reservoir state perturbations $q_n^{(k)}$. If D_n is the derivative of f with respect to u at (r_{n-1}, u_{n-1}), then the affect of perturbations applied only at the most recent time step can be approximated linearly as $r_n^{(k)} - r_n \approx \varepsilon D_n p_n^{(k)}$. Let $P_n = \left[p_n^{(1)} \cdots p_n^{(K)}\right]$, where $p_n^{(k)}$ is the k^{th} perturbation added at time index n, and let $Q_n = \left[r_n^{(1)} - r_n \cdots r_n^{(k)} - r_n\right]$. Then $Q_n \approx \varepsilon D_n P_n$ for small ε, and

$$\frac{1}{K}\sum_{n,k}\left\|Wq_n^{(k)}\right\|^2 = \frac{1}{K}\sum_n \|WQ_n\|^2 \approx \frac{\varepsilon^2}{K}\sum_n \|WD_nP_n\|^2. \tag{16}$$

The effect of using multiple perturbations can be approximated by adding $(\varepsilon^2/K)\sum_n D_n P_n P_n^T D_n^T$ to the usual regularization matrix $\Gamma\Gamma^T$. For random perturbations $p_n^{(k)}$ with mean zero (as were used in our previous work), the expected value of $P_n P_n^T$ is K times the co-variance matrix of the perturbations. In the large K limit, we may use this covariance matrix instead of actual

perturbations. For independent Gaussian perturbations with standard deviation 1, the resulting regularization matrix is $\varepsilon^2 \sum_n D_n D_n^T$.

The effect of perturbations applied over a finite number of time steps may also be linearly approximated in a more complicated way. If we let E_n be the derivative of f with respect to r at (r_{n-1}, u_{n-1}), then the expected regularization from m perturbation steps is:

$$\varepsilon^2 \sum_n \Big[\Big(\sum_{j=n-m+1}^{n} \Big(\prod_{i=j+1}^{n} E_i \Big) D_j \Big) \Big(\sum_{j=n-m+1}^{n} \Big(\prod_{i=j+1}^{n} E_i \Big) D_j \Big)^T \Big]. \tag{17}$$

We note that if we find that the effect of random perturbations can be mimicked in a non-random way by an appropriate regularization term, then maybe a regularization term determined by a simpler method would also work well.

4 A Scheme for Optimizing Machine Learning Hyperparameters for Predicting the Climate of Non-Stationary Dynamical Systems

Accurately predicting the climate of a non-stationary dynamical system using machine learning (ML) requires that the ML-predicted motion (1) accurately capture the dynamics of the system of interest and (2) be stable for long periods of time (i.e., much longer than τ_s, the characteristic time-scale over which the system state varies). Both of these qualities depend not only on the training protocol, as discussed in the previous section, but also depend sensitively on the choice of ML hyperparameters (e.g., parameters of the ML model which are not learned but instead chosen a priori). As a result, in order to use ML to predict the climate of non-stationary dynamical systems, we require a scheme for choosing the set of optimal hyperparameters which reflect both of the above goals.

We propose the following scheme and describe details, where applicable, using reservoir computing since that is our choice of machine learning platform. First we partition the measured time series ($\{x(t)\}_{t=-T}^{t=0}$) into 3 disjoint parts: (1) a short validation set, (2) a training set, and (3) a long validation set. The short validation set corresponds to $-T_1 < t \leq 0$, where T_1 is on the order of several (say 20) τ_s. The training data set will correspond to $-T_2 \leq t \leq -T_1$ where the training length $t_t = T_2 - T_1$ is a hyperparameter. The long validation data set will correspond to $-T_3 \leq t < -T_2$, where $(T_3 - T_2)$ will be on the order of many times τ_s (say $200\tau_s$). After training the output weights of the reservoir computer on the training data, we first determine how well this trained reservoir unit can predict the short-term state of the true system by considering the median "valid time" over a set of m_1 predictions. The valid time of a predicted trajectory is obtained by calculating a normalized Euclidean error between the reservoir computer predicted trajectory and the true system trajectory and monitoring where this error crosses a chosen threshold. To calculate the median valid time, we first randomly select m_1 starting points in the short validation set near $t = -T_1$, resynchronize the reservoir computer to a short segment of data (say of length τ_s) starting at each of the randomly selected starting points (i.e., run it in "open-loop") and then predict ("close the loop"). The median of the m_1 valid times is then called the median valid time for the trained reservoir computer used (and for the specific set of hyperparameters). The stability of predictions can be assessed by similarly making m_2 independent predictions over the long validation set, starting near $t = -T_3$. In this case, instead of calculating the valid time of each trajectory, we calculate the Wasserstein distance between the distribution of states of the predicted trajectory and that of the true system. For multivariate systems, we simply compute the Wasserstein distance for each variable and then average them. Furthermore, the integral in calculating the Wasserstein distance is performed over the range of values corresponding to the range of that variable of the true trajectory. The average Wasserstein distance for each set of

hyperparameters is then obtained by averaging over the m_2 predictions. For each set of hyperparameters, we calculate the median valid time (t_v) and the average Wasserstein distance (E_W). We discard all hyperparameter sets which yield unstable predictions from consideration, this can be determined by seeing if the ML-predicted trajectory diverges to infinity, resulting in a numerical overflow error. Then we calculate the following quantity for each hyperparameter set

$$\mathcal{E} = E_W/\text{median}(E_W) - t_v/\text{median}(t_v) \tag{18}$$

where the median() of E_W and t_v is calculated over all hyperparameter sets (excluding those discarded due to unstable prediction). Finally, the set of hyperparameters which minimize $\mathcal{E}$ are taken as the "appropriate" set of hyperparameters.

As described above, we want to obtain hyperparameters which allow the reservoir computer to learn the true dynamics of the system of interest as well as produce predictions which are stable over a period of time much longer than τ_s. The first of these is accomplished by evaluating the median valid time. The second is accomplished by evaluating the average Wasserstein distance over the long validation set. A reservoir computer prediction becomes unstable when a small perturbation in a direction perpendicular to the manifold on which the learned dynamics are embedded grows, in other words, when the Lyapunov exponent in that direction is positive. The rate at which this small perturbation grows determines how long it takes for the predicted trajectory to become unstable (i.e., move far away from the neighborhood of the manifold which contains the learned dynamics of the system of interest). So in the case where the system of interest is chaotic, if we choose the long validation sequence length to be approximately $n\tau_s \approx n/\lambda_L$, where λ_L is the most positive Lyapunov exponent of the system of interest, a reservoir prediction which consistently (over all m_2 predictions for m_2 reasonably large) remains stable over the entire length of the long validation set indicates that the most positive Lyapunov exponent of the reservoir computer, for that choice of hyperparameter set, is at most on the order of $1/(n\tau_s)$. This implies that we can expect a typical reservoir prediction to then be stable for at least periods of time on the order of $n\tau_s$. To ensure that our choice of hyperparameters allows the reservoir to simultaneously learn the true dynamics of the system of interest well and to generate predictions which remain stable, we combine the median valid time and average Wasserstein distance metrics as done in Eq. 18. The order of the training data set and the short and long validation sets are chosen so that we perform training and the median valid time tests closest to the present time, near $t = 0$, so as to most accurately capture dynamics which are most relevant for prediction (since we consider non-stationary systems).

References

[1] Alexander Wikner, Jaideep Pathak, Brian Hunt, Michelle Girvan, Troy Arcomano, Istvan Szunyogh, Andrew Pomerance, and Edward Ott. Combining machine learning with knowledge-based modeling for scalable forecasting and subgrid-scale closure of large, complex, spatiotemporal systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(5):053111, 2020.

[2] Troy Arcomano, Istvan Szunyogh, Alexander Wikner, Jaideep Pathak, Brian R Hunt, and Edward Ott. Hybrid approach to atmospheric modeling that combines machine learning with a physics-based numerical model. *Earth and Space Science Open Archive*, page 37, 2021.