Data-Driven Super-Parameterization Using Deep Learning: Experimentation With Multiscale Lorenz 96 Systems and Transfer Learning

Presenter: Wenlin Li

Outline

1. Background and Introduction
2. The Multiscale Lorenz 96 System and Data(Lorenz 96 System, case 1 and case 2)
3. Methodologies/Models (DNS, HR, PLR, SP, DD-P, DD-SP, Fully-DD)
4. Results
5. Conclusion and Next Steps

Background and Introduction

To make weather and climate models **computationally affordable**, small-scale processes are usually represented in terms of the large-scale, explicitly resolved processes using **physics-based/semi-empirical parameterization schemes**. Another approach, computationally more demanding but often more accurate, is **Data-Driven (DD)**.

The Multiscale Lorenz 96 System and Data

1.Introduce the Lorenz ’96 System

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - \frac{hc}{b}\Sigma_j Y_{j,k} \tag{2}$$

$$\frac{dY_{j,k}}{dt} = -cbY_{j+1,k}\left(Y_{j+2,k} - Y_{j-1,k}\right) - cY_{j,k} + \frac{hc}{b}X_k - \frac{he}{d}\Sigma_i Z_{i,j,k} \tag{3}$$

$$\frac{dZ_{i,j,k}}{dt} = edZ_{i-1,j,k}\left(Z_{i+1,j,k} - Z_{i-2,j,k}\right) - geZ_{i,j,k} + \frac{he}{d}Y_{j,k} \tag{4}$$

X represents the large scale but slow process (like the long wave in the atmosphere). F is the force term, in which is external input of influence that drives the system and changes its behavior. And Y represents the small scale but fast process, like the short wave in the atmosphere or the Turbulence. Z represents Gravity Waves (smaller than X and Y)

2.Case 1 and Case 2

Case 1: Scale Separation Between X and Y.

In this case, b = c = e = d = g = 10 and h = 1. In Case 1, there is scale separation between X and Y, which means that the variations in X are relatively slow compared to the variations in Y. (For example, atmospheric convection(Y) (such as **cloud formation** and dissipation) may change on timescales of minutes to hours, while atmospheric circulation(X) (such as **wind** and pressure patterns) may change on timescales of days to weeks.)

Case 2: No Scale Separation Between X and Y

In this case, b = e = d = g = 10, c = 8, and h = 0.5. In Case 2, there is no scale separation between X and Y, indicating that both X and Y vary on similar time scales.

Fourth-order Runge-Kutta (RK4):

The formula for the fourth-order Runge-Kutta method is given by:

$y_1 = y_0 + (⅙) (k_1 + 2k_2 + 2k_3 + k_4)$

Here,$k_1 = hf(x_0, y_0)$, $k_2 = hf[x_0 + (½)h, y_0 + (½)k_1]$, $k_3 = hf[x_0 + (½)h, y_0 + (½)k_2]$, $k_4 = hf(x_0 + h, y_0 + k_3)$

Example: Consider an ordinary differential equation dy/dx = x^2 + y^2, y(1) = 1.2. Find y(1.05) using the fourth order Runge-Kutta method:

Given, dy/dx = x^2 + y^2, y(1) = 1.2 So, f(x, y) = x^2 + y^2. x0 = 1 and y0 = 1.2 Also, h = 0.05(step sizeΔt) Let us calculate the values of k_1, k_2, k_3 and k_4.

$k_1 = hf(x_0, y_0)$

$= (0.05) [x_0^2 + y_0^2]$

$= (0.05) [(1)^2 + (1.2)^2]$

$= (0.05) (1 + 1.44)$

$= 0.122$

$k_2 = hf[x_0 + (½)h, y_0 + (½)k_1]$

$= (0.05) [f(1.025, 1.261)]$

$= (0.05) [(1.025)^2 + (1.261)^2]$

$= 0.1320$

$k_3 = hf[x_0 + (½)h, y_0 + (½)k_2]$

$= (0.05) [f(1.025, 1.266)]$

$= 0.1326$

$k_4 = hf(x_0 + h, y_0 + k_3)$

$= (0.05) [f(1 + 0.05, 1.2 + 0.1326)]$

$= (0.05) [f(1.05, 1.3326)]$

$= 0.1439$

By RK4 method, we have;$y_1 = y_0 + (⅙) (k_1 + 2k_2 + 2k_3 + k_4)$

$y_1 = y(1.05) = y_0 + (⅙) (k_1 + 2k_2 + 2k_3 + k_4)$ By substituting the values of y_0, k_1, k_2, k_3 and k_4, we get: $y(1.05) = 1.2 + (⅙) [0.122 + 2(0.1320) + 2(0.1326) + 0.1439] = 1.2 + (⅙) (0.122 + 0.264 + 0.2652 + 0.1439) = 1.2 + (⅙) (0.7951) = 1.2 + 0.1325 = 1.3325$

Methodology/Model

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - \frac{hc}{b}\Sigma_j Y_{j,k} \quad (2)$$

$$\frac{dY_{j,k}}{dt} = -cbY_{j+1,k}\left(Y_{j+2,k} - Y_{j-1,k}\right) - cY_{j,k} + \frac{hc}{b}X_k - \frac{he}{d}\Sigma_i Z_{i,j,k} \quad (3)$$

$$\frac{dZ_{i,j,k}}{dt} = edZ_{i-1,j,k}\left(Z_{i+1,j,k} - Z_{i-2,j,k}\right) - geZ_{i,j,k} + \frac{he}{d}Y_{j,k} \quad (4)$$

1.(Base)Direct Numerical Simulation(DNS, Sample data):

- Method: Numerical integration of all equations(Equations 2–4) using the RK4 scheme.
- Time Step: Δt=0.005, dictated by the high-frequency variables, Y and Z.
- Purpose: Represents the "truth" model. All scales are resolved, and no equation is simplified or ignored.
- Usage: Used as a benchmark for training, validation, and testing of other models.

2.High-Resolution (HR):

- Method: Numerical integration of Equations 2 and 3 using the RK4 scheme.
- Modification: Ignores Variable Z and the last term in Equation 3, only focus X and Y (Equations 2 and 3).
- Purpose: A computationally cheaper model compared to DNS (roughly 8 times less expensive).
- Note: Still too expensive for practical use.

3.Parameterized Low-Resolution(PLR):

Parameterization: The impact of Y on X is parameterized as a function Up(X). In reality, this function represents the influence of Y and is incorporated into the evolution equation for X (Equation 5).(Only focus on X)

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - \frac{hc}{b}\Sigma_j Y_{j,k} \qquad (2)$$

The parameterization function **Up(X)** employs a **stochastic parameterization scheme** (Equation 6). In this equation, **U_det(X)** is a **deterministic function represented by a fourth-order polynomial** (Equation 7), and **the coefficients of Equation 7 are determined separately for Case 1 and Case 2 using a fourth-order polynomial fit on DNS data.**The other component, **η(t), is a random process** (Equation 8) that introduces stochasticity to simulate effects that cannot be captured by the deterministic function U_det(X).

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - U_p(X_k) \qquad (5)$$

$$U_p(X) = U_{det}(X) + \eta(t) \qquad (6)$$

$$U_{det}(X) = a_0 + a_1 X + a_2 X^2 + a_3 X^3 + a_4 X^4 \qquad (7)$$

$$\eta(t) = \phi\eta(t - \Delta t) + \sigma_\eta (1 - \phi^2)^{\frac{1}{2}} z(t) \qquad (8)$$

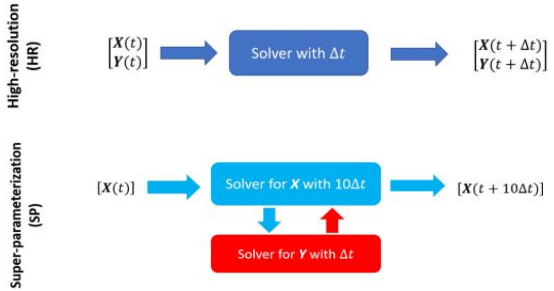
Here ϕ is the lag-1 autocorrelation and σ_η is the standard deviation of the stochastic time series, η, computed following the methodology described in Arnold et al. (2013). z(t) is white noise with unit standard deviation.

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - \frac{hc}{b}\Sigma_j Y_{j,k} \quad (2)$$

$$\frac{dY_{j,k}}{dt} = -cbY_{j+1,k}\left(Y_{j+2,k} - Y_{j-1,k}\right) - cY_{j,k} + \frac{hc}{b}X_k - \frac{he}{d}\Sigma_i Z_{i,j,k} \quad (3)$$

4.Super-Parameterization (SP):

- Method: Uses a super-parameterization approach. Integrates Equation 2 using the RK4 scheme with a time step of 10Δt. Then, uses the output to integrate Equation 3 with a time step of Δt.
- Modification: Ignores variable Z.
- Purpose: Adheres to the super-parameterization philosophy for multiscale systems. Slightly cheaper than the HR model.



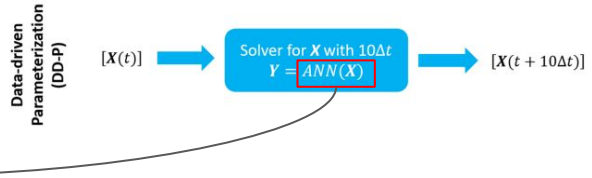

```
for i in range(0,int(t_max/dt_x)-1):
  x_vec, y_mat = x_step(x_vec, y_mat, dt_x)
  x_store[int(j*2000)+i,:]=x_vec
```

```
for i in range(0,int(t_max/dt_x)-1):
  y_mat, y_final = y_step(x_vec, y_mat, dt_y, dt_x)
  x_vec = x_step(x_vec, y_final, dt_x)
  x_store[int(j*200)+i,:]=x_vec
```

5.Data-Driven Parameterization (DD-P):

- Method: Similar to the PLR model but uses a data-driven representation of Up obtained from an artificial neural network (ANN, 8 hidden layers (Full connection layer), activation function: Relu).
- Training: Trained with 1M(10^6) sequential pairs of X and Y that are sampled every Δt from the DNS data.
- Purpose: Represents models that use machine learning methods to learn data-driven representations of subgrid processes.
- Cost: Same as PLR.

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - U_p(X_k) \quad (5)$$

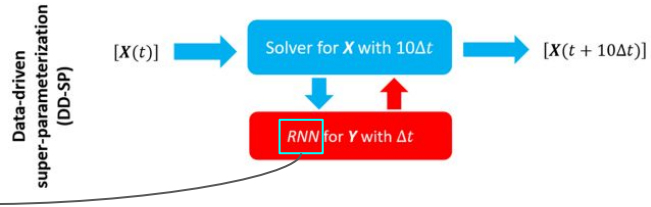


6.Data-Driven Super-Parameterization (DD-SP):

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - U_p(X_k) \quad (5)$$

- Method: Similar to the SP model but integrates Equation 3 using a recurrent neural network (RNN, They use a gated recurrent unit (GRU) as the RNN) instead of a numerical solver.
- Training: Trained with 1M(10^6) sequential pairs of X and Y that are sampled every Δt from the DNS data.
- Purpose: Aims to reduce computational cost by replacing numerical integration with data-driven integration.
- Cost: Roughly 88 times cheaper than SP and has the same cost as PLR and DD-P.

$$\frac{dX_k}{dt} = X_{k-1}\left(X_{k+1} - X_{k-2}\right) - X_k + F - U_p(X_k) \quad (5)$$



7.Fully Data-Driven(DD) (The difference from DD-SP is that Fully DD only considers X and uses the historical value of X to predict X):

- Method: Uses an RNN trained on X to predict the spatiotemporal evolution of X from an initial condition.
- Training: Trained on 1M(10^6) sequential values of X(t) sampled at every 10Δt from the DNS data.
- Purpose: A completely data-driven model for predicting the system's evolution.

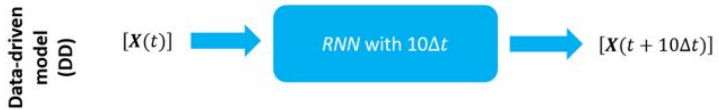


Table 1
Normalized Computational Cost of Each Model

Model	Direct numerical simulation (DNS)	High-resolution (HR)	Super-parameterized (SP)	Data-driven super-parameterized (DD-SP)	Fully data-driven (DD)	Data-driven parameterized (DD-P)	Parameterized Low-resolution (PLR)
Cost	1,000	123.3	111.0	1.4	≪1	1.4	1.4

Note. Cost is evaluated as the number of equations (of X and if applicable, Y and Z) that need to be solved numerically to compute $X(t+10\Delta t)$ from $X(t)$. Therefore, the reported costs account for differences in the time step size (Δt vs. $10\Delta t$) used in different models. We emphasize that the costs for solving the equations of small-scale variables (Y and Z) are also included if solving these equations numerically is part of the model, for example, in DNS, HR, and SP. Costs are normalized such that the cost of DNS is 1,000. We have assumed that the computational cost of running a *trained* neural network is negligible compared to the computational cost of a numerical solver; therefore, the estimated cost of DD is negligible compared to the cost of other models and the estimated cost of DD-P and DD-SP is only due to the numerical part (and thus the same as PLR).

Results

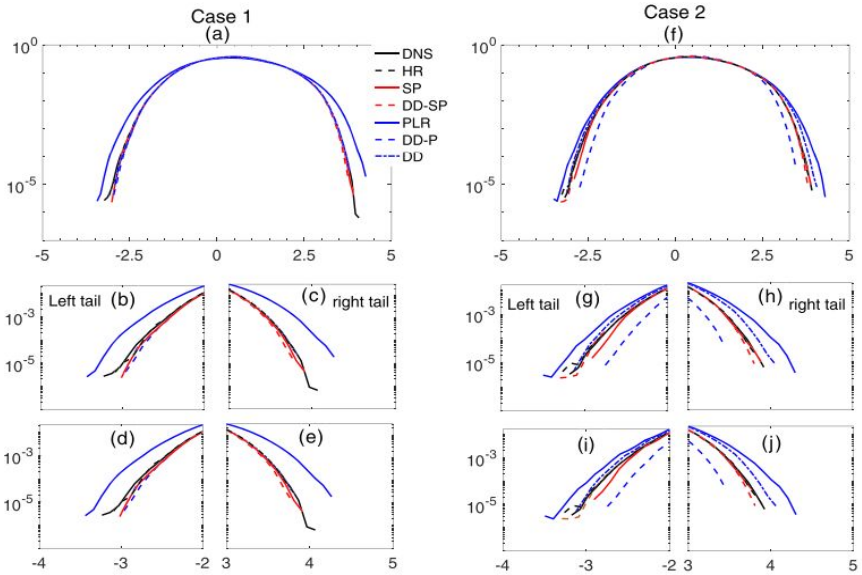
1. Short-Term Spatiotemporal Forecasting:

- **Without noisy initial conditions(Precise Initial Conditions)**
  - Experiment: The prediction accuracy of all models for Cases 1 and 2 is compared.
  - Analysis:
    - HR performs the best among purely **numerical models**, with SP following closely behind. PLR has the shortest prediction horizon.(Accuracy: **HR(only X and Y) > SP(only X and Y but interaction) > PLR(X and Up(X))**)
    - HR and SP are computationally more expensive than PLR.
    - The data-driven models (DD-P and DD-SP) with low computational costs perform significantly better than PLR in Case 1.
    - DD-SP performs slightly better than DD-P.
    - In Case 2, DD-P's performance decreases compared to PLR while DD-SP still outperforms PLR.
    - The fully DD model's accuracy is similar to that of the PLR model.(Accuracy: **DD ~= PLR**)

- **With noisy initial conditions(Noisy Initial Conditions)**
  - Experiment: The experiment was repeated but with noisy initial conditions.
  - Analysis:
    - As expected, the prediction horizons for all models decrease, but more so for the more accurate models.
    - **DD-SP**'s advantages become even more apparent, especially when compared to PLR and DD-P.
    - In terms of computational costs, **DD-SP** provides **performance similar** to the more **expensive SP** model.

2. Long-Term Statistics:

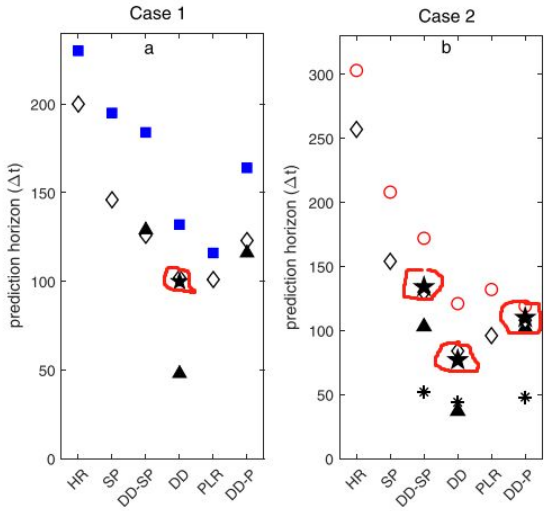
- Experiment: The ability of each model to reproduce long-term system statistics was examined.
- Analysis:
  - For Case 1, all models **except PLR** accurately reproduce the system's long-term statistics.
  - For Case 2, only **DD-SP** and **DD** models accurately reproduce the system's statistics, particularly at the **tails** of the probability density function (PDF).
  - Overall, DD-SP is the best low-cost model in reproducing the **long-term statistics** of the system.

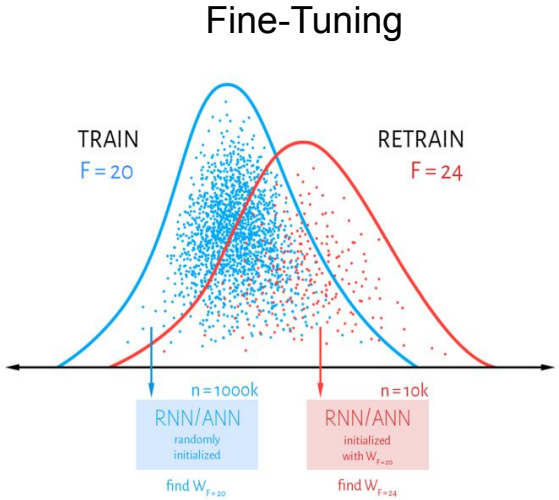


PDFs, focus on tails, which involves forecasting extreme weather

3. Generalization to Systems With Higher F: A Transfer Learning Approach

- Experiment: The generalizability of the models was tested by increasing the forcing F from 20 to 24, making the system more chaotic.
- Analysis:
  - Generalization remains a **challenge for most models**. Fully data-driven models (like DD) struggle more than hybrid models (like DD-P and DD-SP).
  - Transfer learning, which utilizes a well-trained model from one system to train on another, can help **bridge the generalization gap**.
  - For Case 1 and 2, **using transfer learning**, the **DD** model's short-term prediction accuracy **improved**, and the generalization gap for DD-SP and DD was closed.





Conclusion

- **DD-SP Framework**: The paper introduced a Data-Driven Super-Parameterization (DD-SP) model. This model combines numerical integration for large-scale processes and data-driven methods for small-scale processes. The use of deep learning techniques makes this model as cost-effective as low-resolution models.
- **Performance**: The DD-SP model outperforms the Parameterized Low-Resolution (PLR) model in predicting both **short-term changes and long-term statistics**. Moreover, it's found to be computationally **more efficient** than the **Super-Parameterization (SP)** model while providing almost similar accuracy.
- Generalization: The paper evaluates how well models trained on one system (with a specific forcing) can predict the behavior of another system with different characteristics. While all models **generalize well for Case 1**, but they **struggle in Case 2**. However, **using transfer learning** (re-training the models on a small dataset from the new system) helps in **improving the generalization**. The article emphasizes that just examining the Probability Density Functions (PDFs) isn't sufficient to evaluate generalization, especially when focusing on the **tails** of these PDFs.

Next Steps(Limitation)

- **Future Testing**: It's suggested that further tests are needed on **more complex** systems to determine if DD-SP and transfer learning can enhance representations in weather and climate models. The **two-layer quasi-geostrophic model(more complex than Lorenz 96)** is identified as the next system for testing.
- **Instabilities in Hybrid Models**: While the DD-P and DD-SP models discussed are stable, other **hybrid models** have reported numerical **instabilities** in **more complex systems**. It's suggested that these instabilities need thorough **doing more research** when applying DD-SP to more intricate systems.
- **Stochastic Parameterization**: The paper suggests the future exploration of stochastic parameterization, especially for handling irreducible model uncertainty. Techniques like **Generative Adversarial Networks (GANs)** could be useful in this context.