# Assessing Machine Learning Techniques as Emulators for Simple Physics in the Community Atmosphere Model

**Garrett C. Limon** & Christiane Jablonowski

**University of Michigan, Ann Arbor, USA**

**Department of Climate and Space Sciences and Engineering**

**CESM Atmosphere & Whole Atmosphere & Chemistry-Climate Working Groups Meeting**

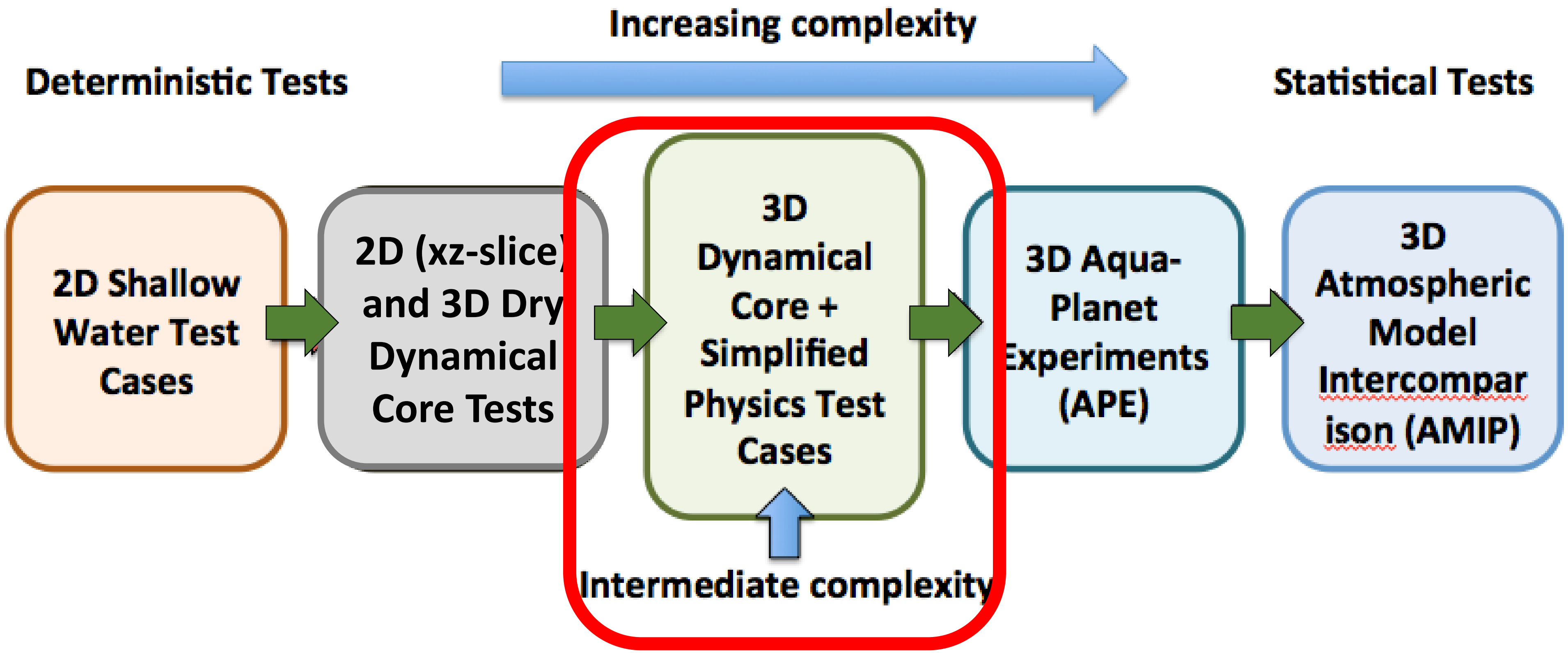
**Wednesday, February 10th, 2021**

# Overarching Questions

- Can Machine Learning (ML) methods replace or augment physical parameterizations in atmospheric GCMs?
- How well can ML methods capture the parameterization scheme?
  - Physical realism
  - Dynamic range
  - Computational complexity/efficiency & data availability
- How does the ML performance depend on the data selection & preparation, ML technique, and the architecture/hyperparameter choices?
  - We utilize an ML tuning tool called Sherpa
- How can we embed physical constraints?

Answer some of these questions with the help of a GCM model hierarchy

# Bridging the Gap:
# Model Hierarchy with Increasing Complexity




**GCMs with Simplified Physics (for climate time scales)**

- Dry Held-Suarez test (Held and Suarez, BAMS 1994)
- Moist version of the Held-Suarez test (Thatcher and Jablonowski, GMD 2016)

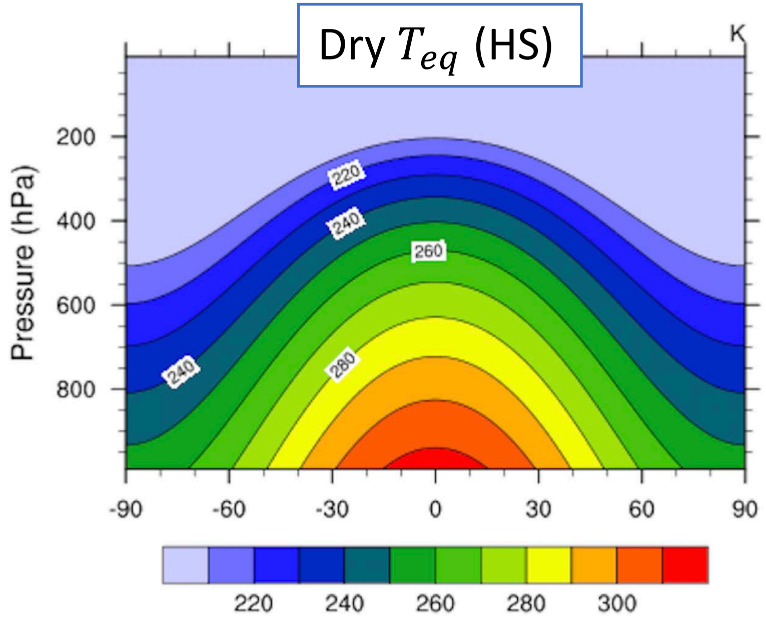
# Idealized Model Setups: Dry Held-Suarez (HS)

- Simplified HS forcings are Rayleigh friction and a Newtonian temperature relaxation:

Linear: $$\frac{\vec{v}_h}{\partial t} = -\frac{1}{k_v(p)}\vec{v}_h$$

$k_v$ and $k_T$ are spatially-dependent relaxation coefficients

$$\frac{\partial T}{\partial t} = \boxed{-\frac{1}{k_T(\phi,p)}[T - T_{eq}(\phi,p)]}$$

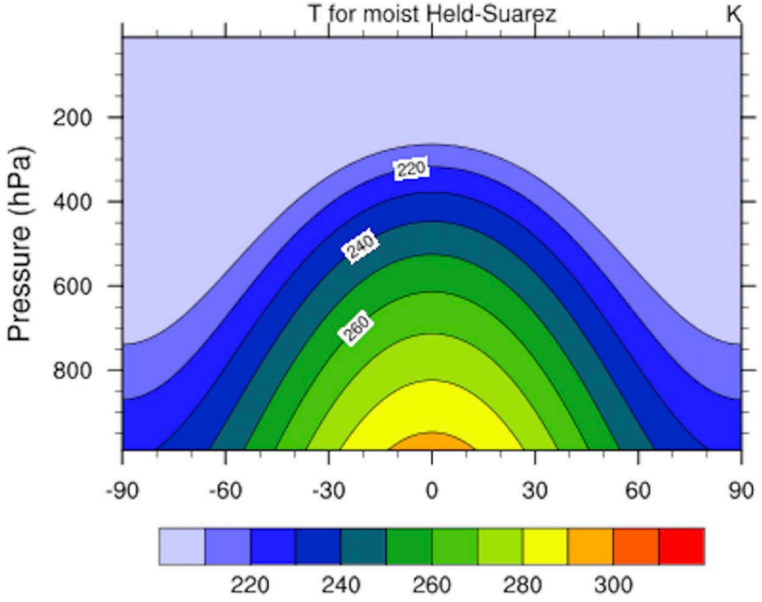



- Focus here: Can ML mimic the physics time tendency of the temperature T ?



Held and Suarez (BAMS 1994)

# Moist Version of the Held-Suarez test (MHS)




- Simplified MHS forcings (moist) are Rayleigh friction, a Newtonian temperature relaxation, PBL mixing (Laplacian), surface fluxes &rain

$$\frac{\vec{v}_h}{\partial t} = -\frac{1}{k_v(p)}\vec{v}_h$$

$$\frac{\partial q}{\partial t} = -C + \text{PBL diffusion} + \text{surface latent heat flux}$$

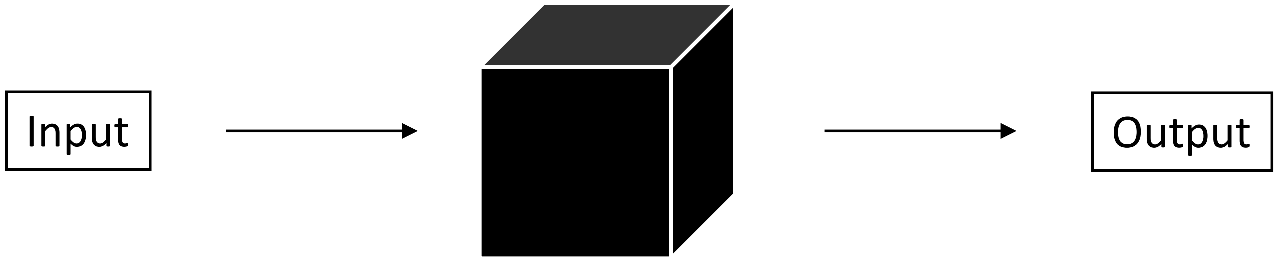
$$\frac{\partial T}{\partial t} = \boxed{-\frac{1}{k_T(\phi,p)}\left[T - T_{eq}(\phi,p)\right] + \frac{L}{c_p}C + \text{PBL diffusion} + \text{surface sensible heat flux}}$$

$$\text{Precipitation rate} = \boxed{\frac{1}{\rho_{water}\, g}\int_0^{p_s} C\, dp}$$

L: Latent heat of vaporization
C: Condensation rate

- Focus here: Can ML (neural network & random forests) mimic the physics time tendency of T? The precipitation rate?

# Machine Learning

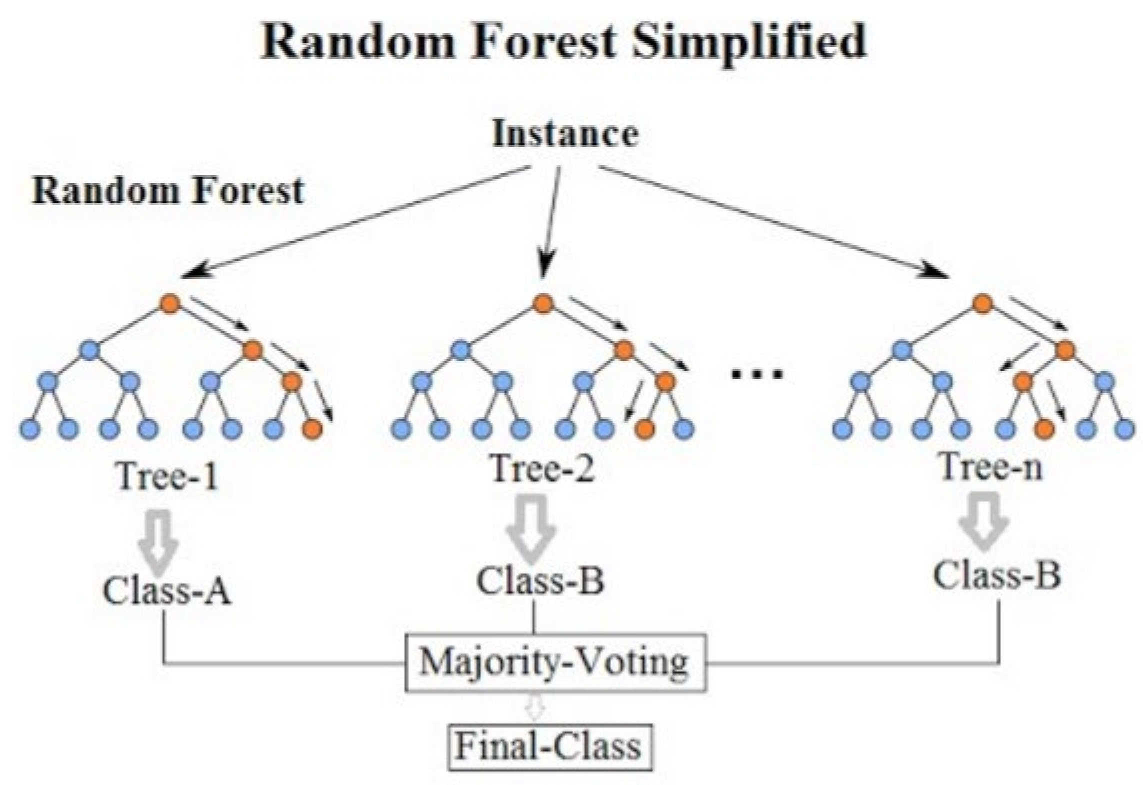



- Determines a functional relationship within a given dataset without being programed to do so
- Requires a lot of data
- Our focus: Neural Network (NN) and Random Forest models
- Built using Keras (TensorFlow) & Scikit-Learn
- Optimized with Sherpa (ML hyperparameter optimization tool)

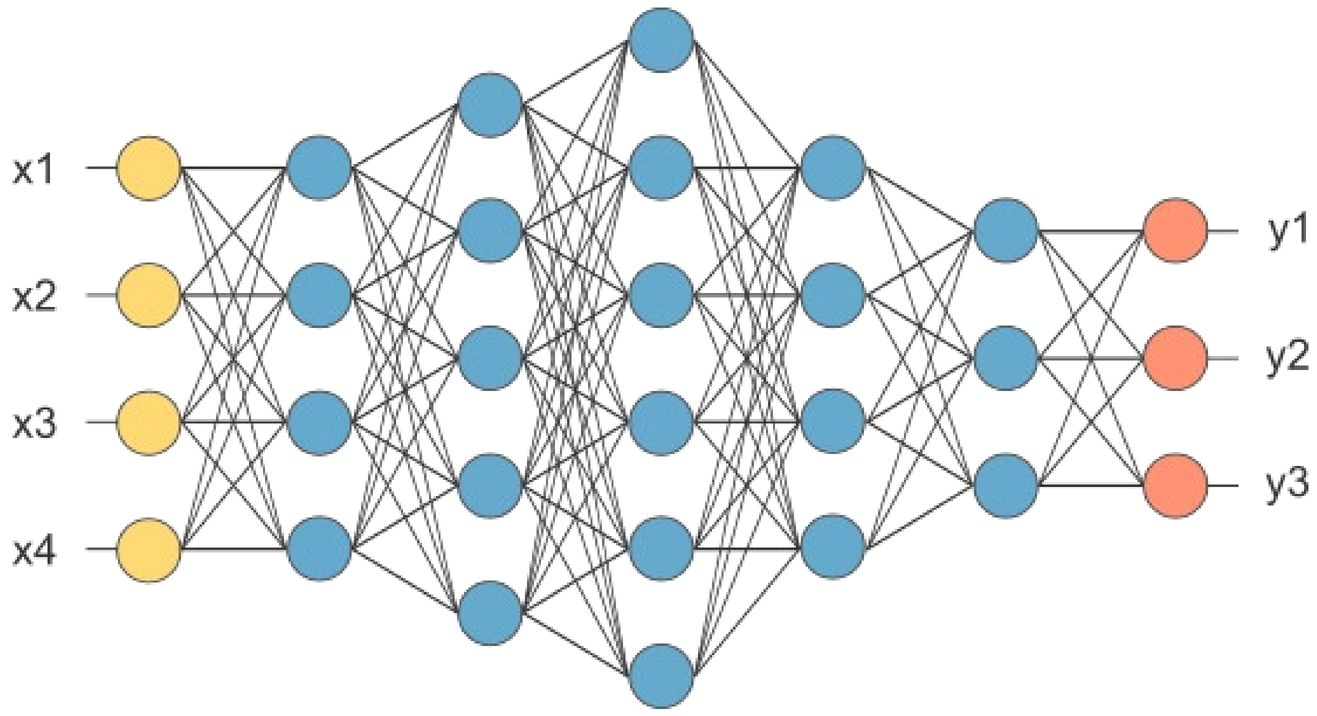
Hertel et al. (2020): https://arxiv.org/abs/2005.04048

https://github.com/sherpa-ai/sherpa

# Random Forests & Neural Networks



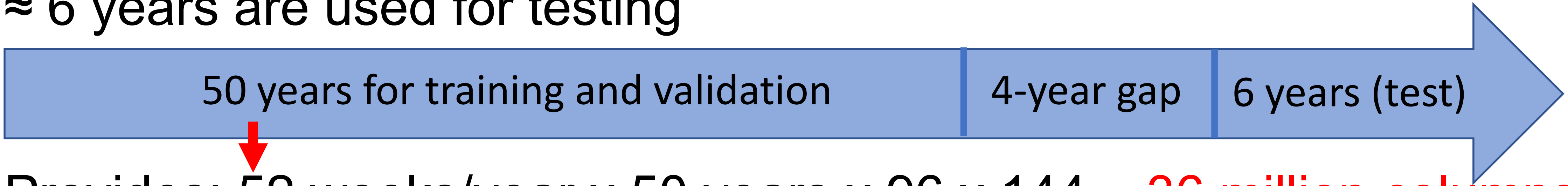

RF image by Venkata Jagannath – [Wikipedia]



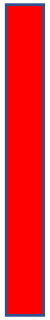

NN image from [WhyAxis]

# GCM Configuration

- NCAR's Community Earth System Model (CESM) version 2.1
- Finite-Volume (FV) CAM6 dynamical core at the resolution 1.9 x 2.5 (96 x 144 grid points) with 30 vertical levels (from now on: 2 degree)
- Model is run for about 60 model years
- Output is collected every week: u, v, $p_s$, T, Q, LHFLX, SHFLX, du/dt, dT/dt, dq/dt, PRECL
- 40 years is used for training, ~10 years for validation (total of 50)
- ≈ 6 years are used for testing




30 levels



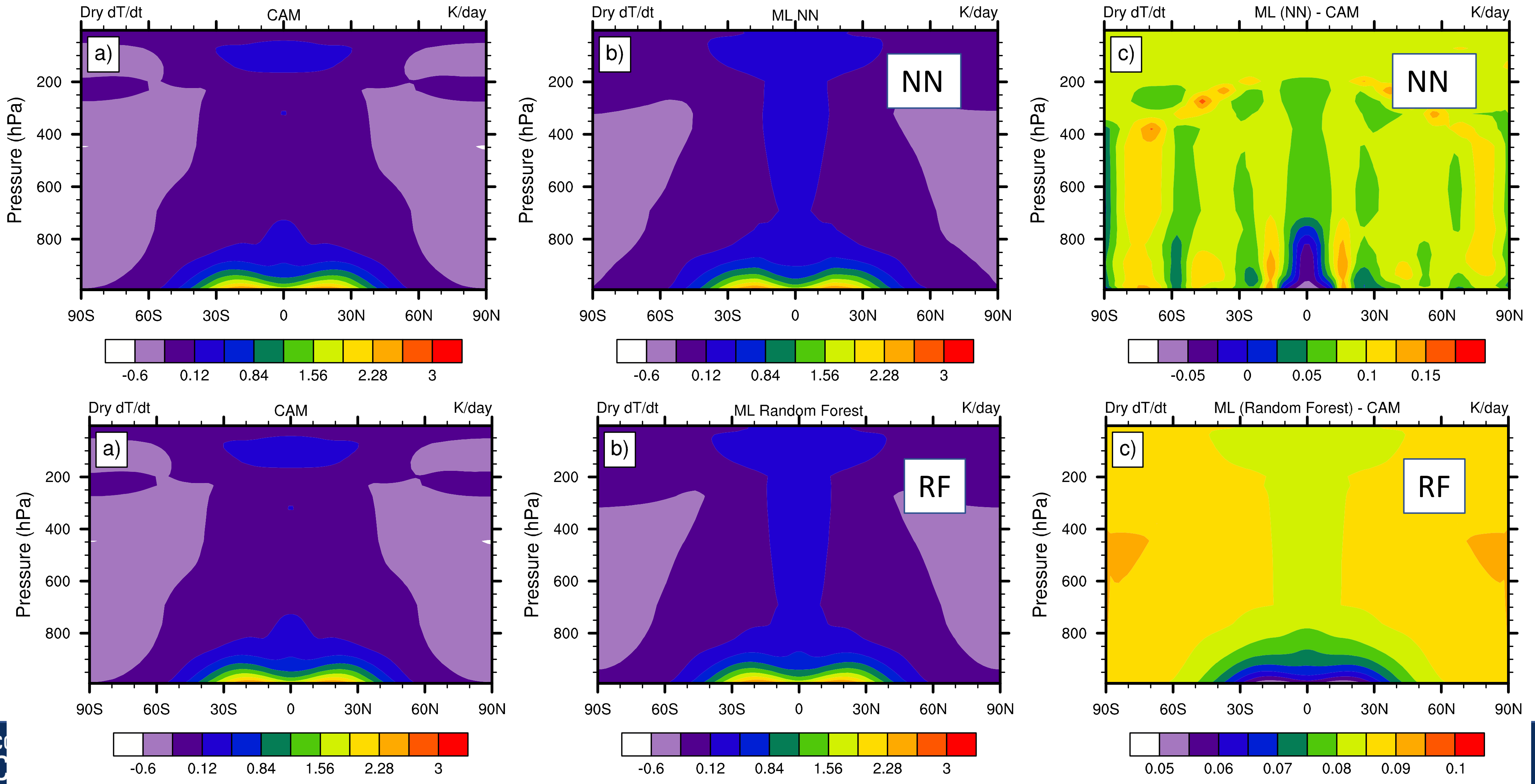
- Provides: 52 weeks/year x 50 years x 96 x 144 = 36 million columns

# Data & Preparation

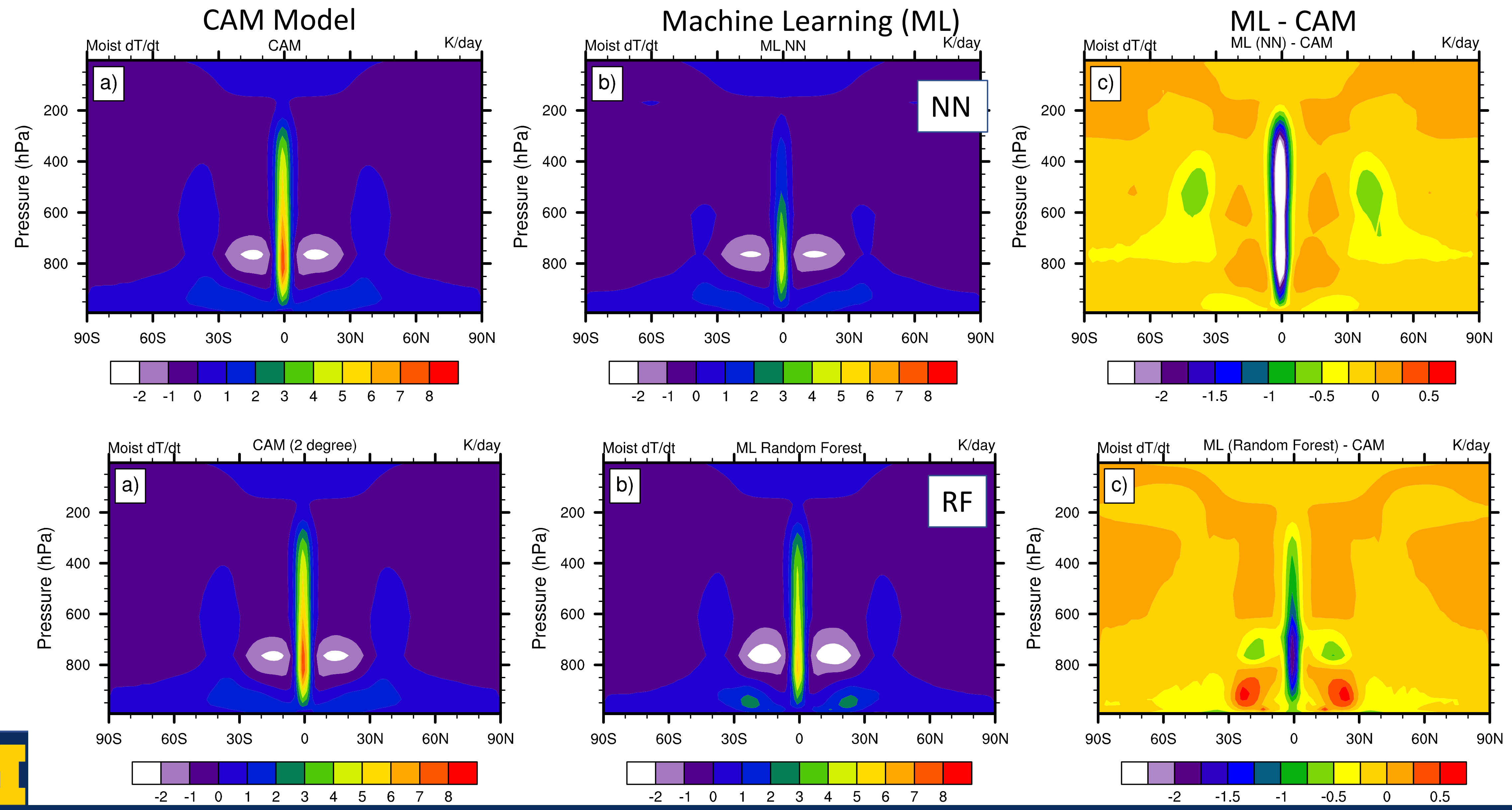
- Training and testing is currently done ‘offline’
- Different ML models are trained for
  - dT/dt (temperature tendency)
  - Precipitation rate
- Data Preparation:
  - Samples grouped into vertical columns:
    - reshape from (time, lev, lat, lon) -> (time*lat*lon, lev, nfeatures)
  - Normalized to be unitarily invariant (subtract the mean at each level, divide by standard deviation), scaled to range $[-1,1]$
  - Data are shuffled during training

# Closer Look at Dry HS Results: dT/dt (HS)

CAM Model | Machine Learning | ML - CAM



# Increasing Complexity: Moist dT/dt (MHS)

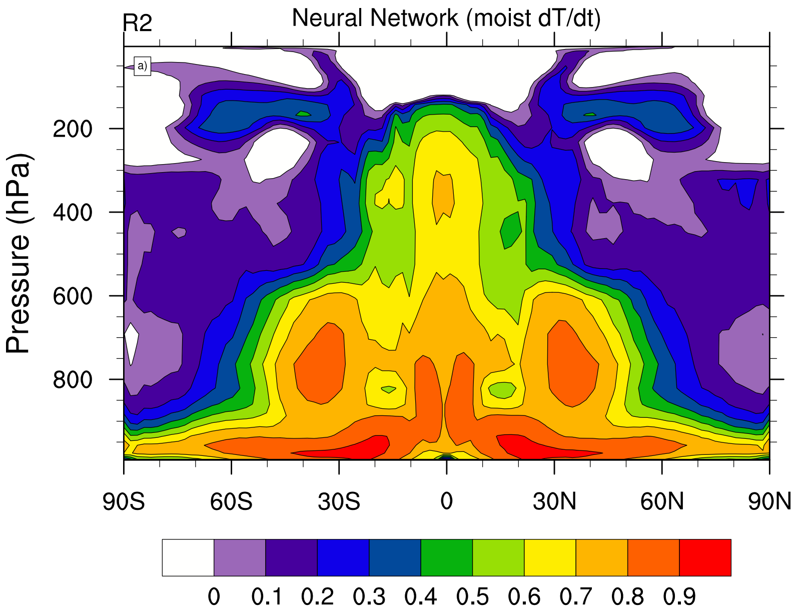


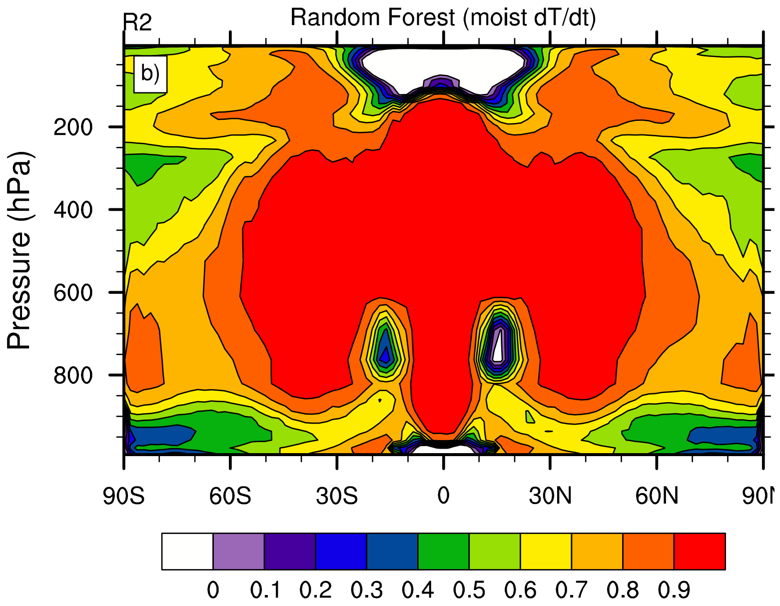

NN

RF

# Analysis of the 2-deg Test Data: Moist dT/dt

- $R^2$ (Coefficient of Determination) assessment reveals problem zones for both
- RF outperforms NN (higher correlations)




Residual sum of squares

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

Variance of the data

High $R^2$ (close to 1) desired.
If $R^2$ negative (white regime): unexplained variance of the ML model exceeds the total variance of the original data.

$$R^2 = 1 - \frac{\sum_{time}\sum_{lon}[\mathrm{CAM}(time,:,:,lon) - \mathrm{ML}(time,:,:,lon)]^2}{\sum_{time}\sum_{lon}[\mathrm{CAM}(time,:,:,lon) - \overline{\mathrm{CAM}}(:,:)]^2}$$

time-mean zonal-mean

# Promising NN performance
# Precipitation (1 degree)

- 30 day animation of the precipitation rate based on 3-hourly data***
- Comparison: CAM (1 deg) versus ML (NN)
- Precipitation bands are captured well by the NN model
- Difficult problem: Precipitation relies on 3D T, p, q and the flow field in order to genera supersaturation (relative humidity > 100%)
- Supersaturation leads to condensation C which needs to be integrated:

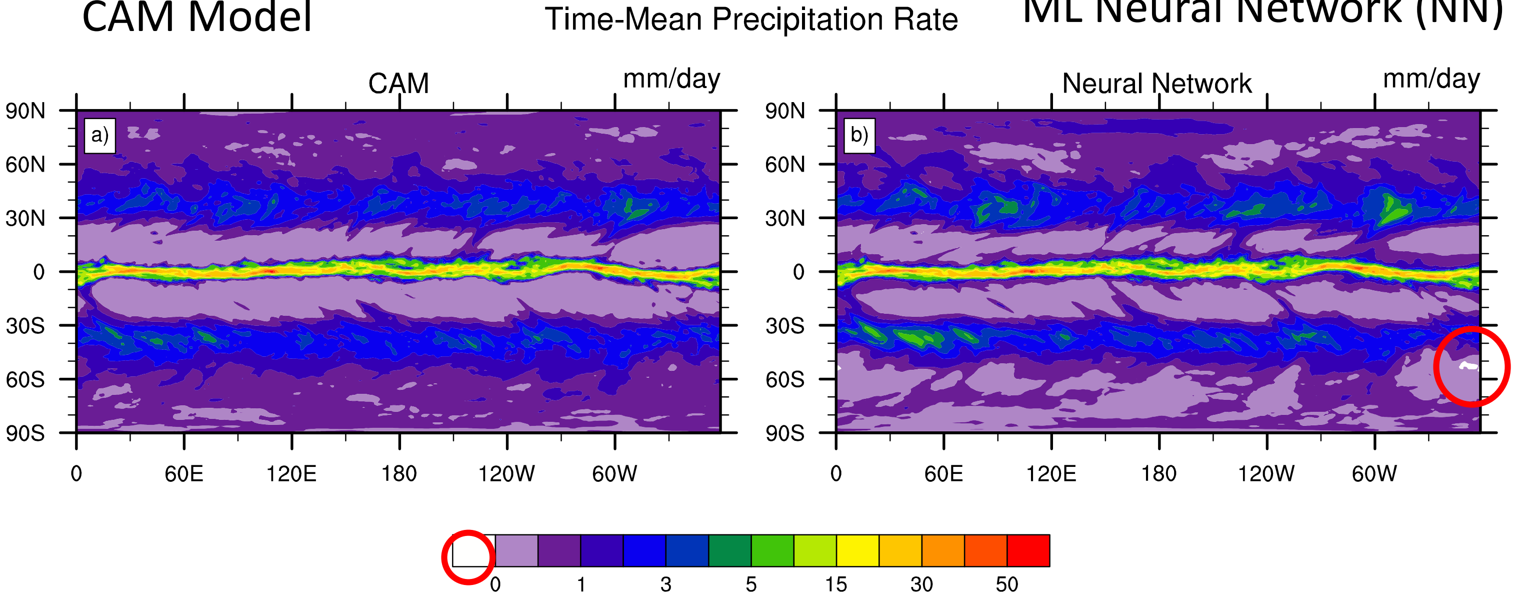
$$\text{Precipitation rate} = \frac{1}{\rho_{water}\, g} \int_0^{p_s} C\, dp$$

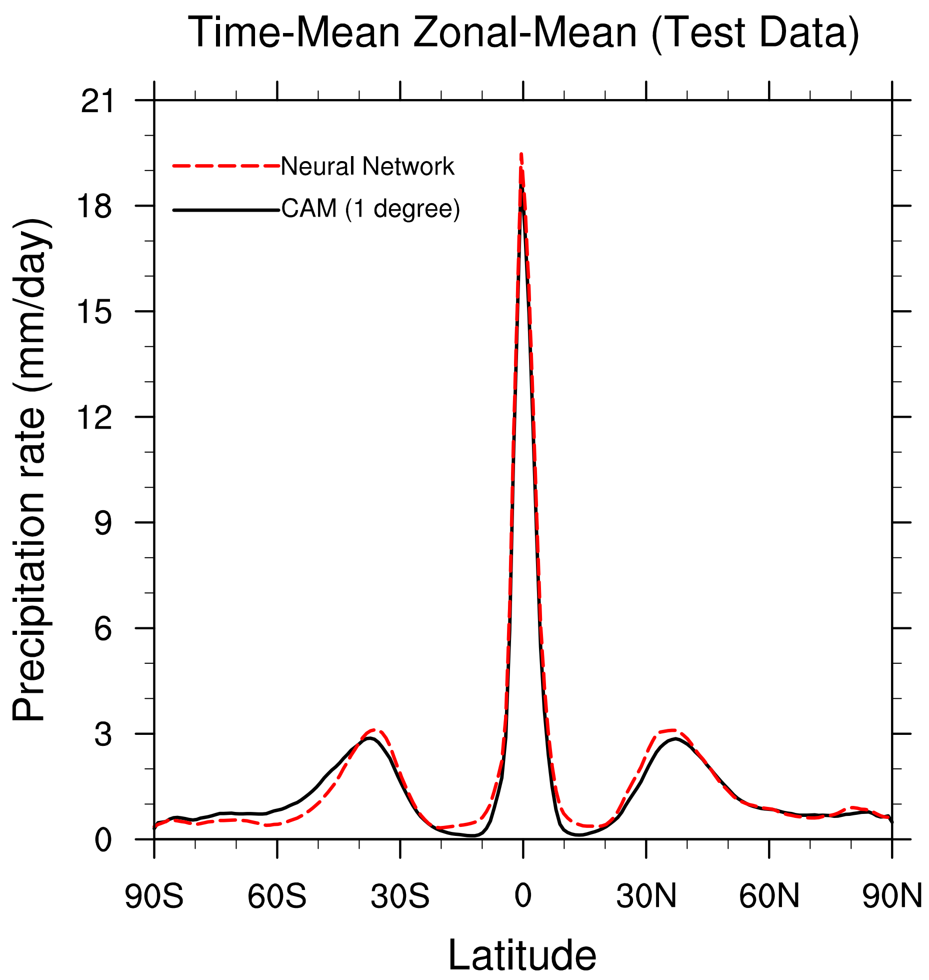
# Precipitation Rate (large-scale condensation, 1 degree)

- Time-mean zonal-means of the test data (68 days):
- CAM and ML (NN) closely resemble each other
- Precipitation peaks are well captured
- However: negative precip rates are possible in NN
- Physical constraints are needed

CAM Model

ML Neural Network (NN)



# Summary

- Machine learning ***can*** emulate these simplified physical parameterizations.
- Both Neural Networks and Random Forests show skill.
- ***Not*** making a claim that our current ML models are optimal (particularly NN).
- Further testing, hyper-parameter tuning and improved data selection are still needed.

# Future Work

- Exploring boosted forests (XGBoost), Convolutional Neural Nets (CNN), and
  other ML techiniques
- Porting these models to be easily coupled to CAM6 (Python-Fortran) & investigate numerical stability in an online mode
- Enforce physical constraints, advance physics-guided ML principles
- Add additional levels of complexity to the hierarchy (aquaplanet, full physics, etc).

# Thank You!

**Contact**

Garrett – glimon@umich.edu

Christiane – cjablono@umich.edu