DLKoopman: A deep learning software package for Koopman theory

Sourya Dey (sourya@galois.com) Eric William Davis (ewd@galois.com)

Created at Galois for DARPA program Symbiotic Design for Cyber-Physical Systems (SDCPS)

pip install dlkoopman

https://github.com/GaloisInc/dlkoopman

Docs: https://galoisinc.github.io/dlkoopman/

dlkoopman

Koopman theory

Data-driven analysis of dynamical systems.

Learn a linear model for a generally nonlinear system only from its snapshots (states).

Encode the states into a different domain using an encoder g. $\boldsymbol{y} = g(\boldsymbol{x})$

Learn the **linear evolution rule** in this domain via the Koopman matrix K. $\boldsymbol{y}_i = \boldsymbol{K}^i \boldsymbol{y}_0$

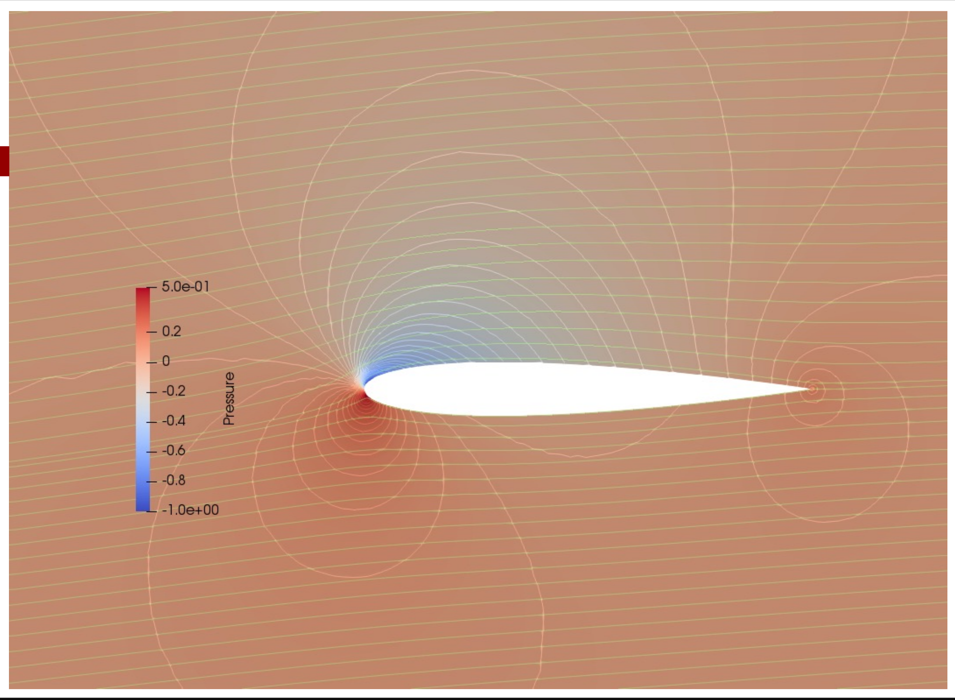
Decode back into the original domain using a decoder g^{-1}. $\boldsymbol{x} = g^{-1}(\boldsymbol{y})$

Example Application

1 m/s flow around the cross-section of a NACA0012 foil in water at 5° angle of attack.

- Pressure contour lines → white
- Streamlines → pale green

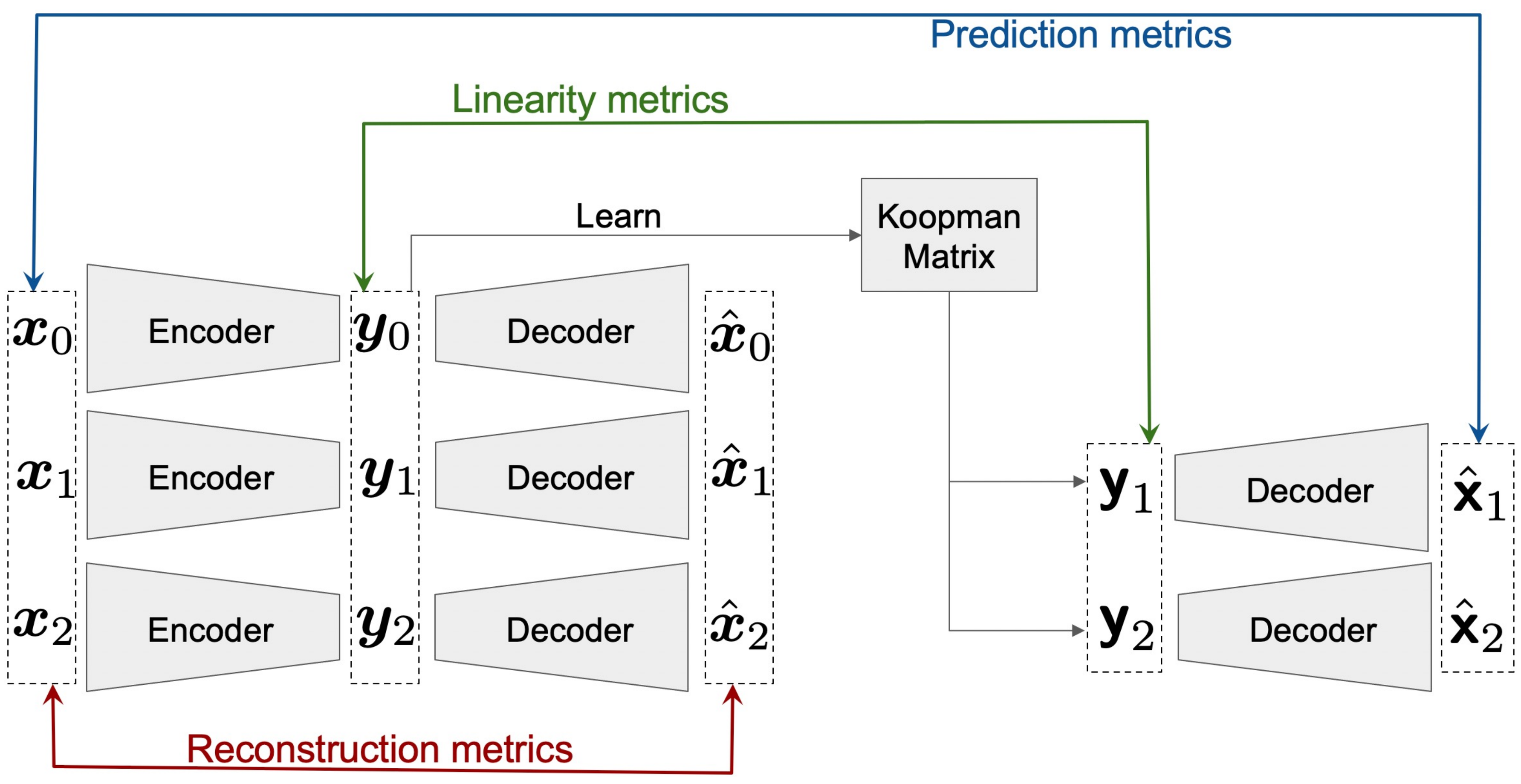
Koopman theory can learn the pressure vector from given index values = angles of attack, then predict it for unknown angles of attack, e.g. 50°, 0.5°, or -2°.



DLKoopman

GaloisInc / dlkoopman Public

Watch 6 Fork 7 Star 31

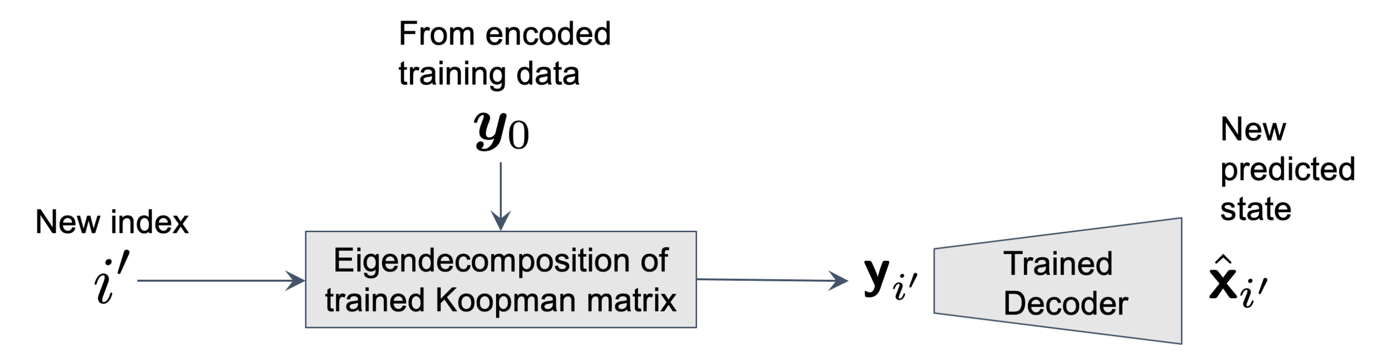


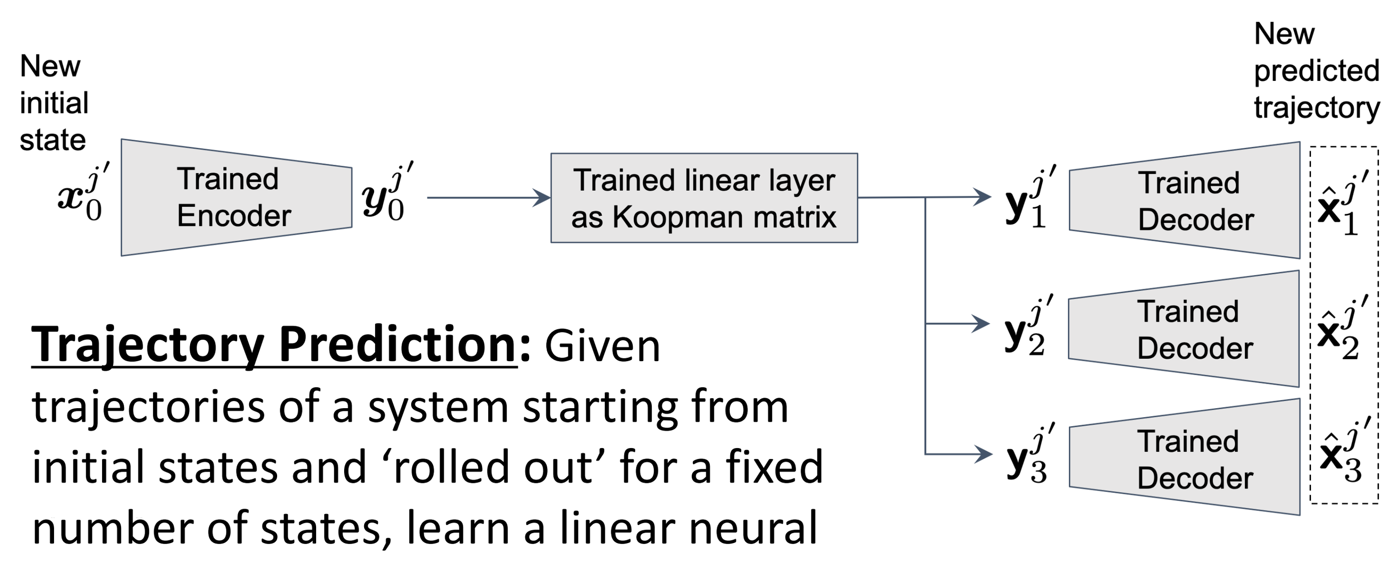
- Reconstruction: How well can the decoder learn the inverse of the encoder?
- Linearity: How good is the linear approximation in the encoded domain?
- Prediction: How good is the overall pipeline at predicting unknown states?

Training loss

$$L = L_{\text{lin}} + \alpha\left(L_{\text{recon}} + L_{\text{pred}}\right) + \beta L_{\text{Autoencoder}} + \gamma L_{\boldsymbol{K}}$$

State Prediction: Given states of a system, learn the Koopman matrix and use its eigendecomposition to predict unknown states in the future, past, or in-between.





Trajectory Prediction: Given trajectories of a system starting from initial states and 'rolled out' for a fixed number of states, learn a linear neural net as the Koopman matrix and use it to predict new trajectories from new initial states.

Example Usage (Python)

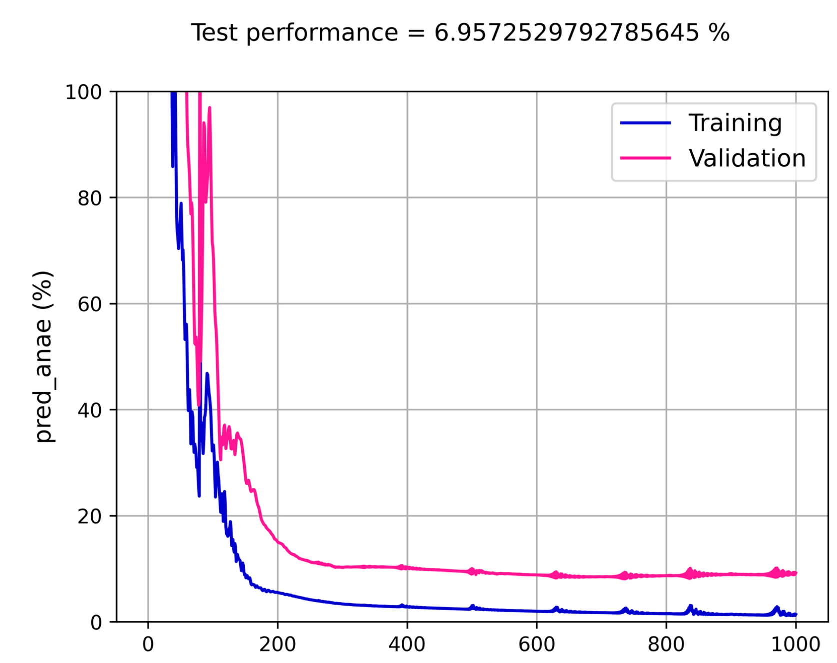
```python
## Define the model
sp = StatePred(
    dh = dh, #StatePredDataHandler instance used to pass data
    rank = 6, #rank of the Koopman matrix K
    encoded_size = 50, #dimensionality of y
    encoder_hidden_layers = [100] #number of neurons
) #above values can be selected via hyperparameter search

## Train the model
sp.train_net(
    numepochs = 1000,
    decoder_loss_weight = 0.1, #alpha in the loss equation
    weight_decay = 1e-5, #beta in the loss equation
    Kreg = 0 #gamma in the loss equation
) #above values can be selected via hyperparameter search

## Perform new predictions
sp.predict_new([3.75,21]) #unknown state indexes
```

Additional Features

Hyperparameter search module to iterate through network sizes and training parameters and select the best configuration(s).

Average Normalized Absolute Error (**ANAE**)

$$\text{ANAE}(\boldsymbol{p}, \boldsymbol{q}) = \text{Avg}_{p_i \neq 0}\left(\frac{|p_i - q_i|}{|p_i|}\right)$$



This material is based upon work supported by the United States Air Force and DARPA under Contract No. FA8750-20-C-0534. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force and DARPA. Distribution Statement A, "Approved for Public Release, Distribution Unlimited."

| galois |