Implementation of Reservoir Computing on KS

April 16, 2019, by Guorui Shen.

reference - [1] & [2], recommended [1], not [2]

[1] Jaideep Pathak, Zhixin Lu, ⋯, Using machine Learning to replicate chaotic attractors and calculate Lyapunov exponents from data.

[2] Jaideep Pathak, Brian Hunt, ⋯. Model-Free Prediction of Large Spatiotemporally chaotic Systems from Data: A Reservoir Computing Approach.

model

$$r(t+\Delta t) = \tanh\left[A \cdot r(t) + W_{in} \cdot u(t)\right]$$

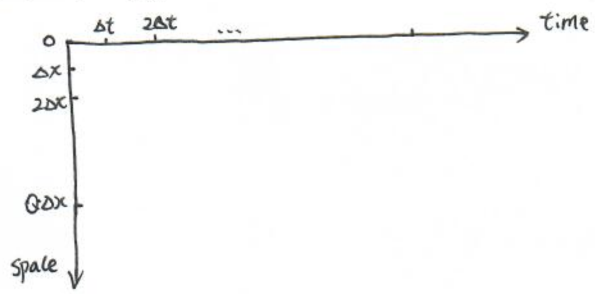
$$v(t+\Delta t) = W_{out}\left(r(t+\Delta t),\ P\right)$$

Kuramoto Sivashinsky (KS)

$$y_t = -y y_x - \left[1 + \mu \cos\left(\frac{2\pi x}{\lambda}\right)\right] y_{xx} - y_{xxxx}$$

$$y = y(x, t)$$

by numerical methods, one can collect its numerical solution at grid points



Reservoir Computing on KS

input:

$$u(t) = \begin{bmatrix} y(\Delta x, t) \\ y(2\Delta x, t) \\ \vdots \\ y(Q \cdot \Delta x, t) \end{bmatrix} \qquad \Delta x = \frac{L}{Q}$$

label:

$$v_d(t) = \begin{bmatrix} y(\Delta x, t) \\ y(2\Delta x, t) \\ \vdots \\ y(Q \cdot \Delta x, t) \end{bmatrix} = u(t)$$

input $u(t)$ ⟶ label $v_d(t+\Delta t)$

$v(t+\Delta t)$ in model try to approximate $v_d(t+\Delta t)$ via

$$\min \sum_t \| W_{out}(r(t), P) - v_d(t) \|_2^2 + \beta \| P \|_2^2$$

note that $u(t-\Delta t) \rightarrow r(t) \longrightarrow v(t)$

Determing unknown parameters

① input to Reservoir: A, W_{in} are initialized at random.

$A \sim$ uniform distribution over interval $[-1, 1]$

$W_{in} \sim$ Uniform distribution over $[-6, 6]$

② Reservoir to Output: W_{out}, P are generall trained, via Gradient Descent, or Ridge, or some other methods.

initial values

$$r(t=0) = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$ or other kind of initialization.

dimension requirements

$$\dim(r(t)) \gg \dim(u(t))$$