SINDy - Autoencoders

SINDy : "Sparse Identification of Non-Linear Dynamics"

Setting : Have a dynamic system that produces real-world <u>measurements</u>

$$x(t) \quad , \quad \frac{dx(t)}{dt} = \dot{x}(t) = \dot{x} \quad , \quad \frac{d^2x(t)}{dt^2} = \ddot{x}(t) = \ddot{x}$$

Reasons why we have x :

- Observation takes place in a <u>different coordinate system</u> (e.g. movement of the planets from earth's perspective)
- Unknown <u>canonical</u> coordinates (coordinates of "true" generative model / system)

Problem :

- Ordinary Differential Equation (ODE) $\ddot{x} = f(x, \dot{x})$ is <u>very complicated</u>

- x, x, $\ddot{x}$ could be very high-dimensional (e.g. a video (W, H, C, T))
  and not interpretable

Solution : introduce a low-dimensional variable $z = \varphi(x)$ that encoder the high-dimensional observations and their dynamics

Why do we want z

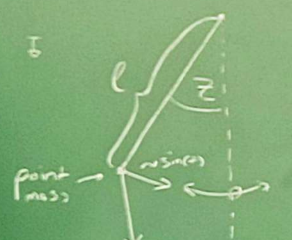
1. Symbolic Regression : Find a simple / "sparse" / "parsimonious" ODE that describes the data with 3 properties:

   - descriptive (small errors at each time-step, and over time : do not want the learned dynamics to diverge)
   - interpretable (symbolic, low-dimensional, "easy" coordinates)
   - parameter efficient (equation with few terms)

2. Efficient Simulation for "surrogate" model : Simulating in x-space may be very expensive

→ simulate in z-space with the learned ODE and decode the simulation back into x-space

Example: Non-Linear Pendulum



$\vec{F}_c = -mg$ "down"

Derivation of the Pendulum-ODE

- Torque $\vec{\tau} = \vec{\ell} \times \vec{F}_c$ with $\vec{\ell}$ from pivot to point mass
- displacement by angle z

→ torque becomes $\tau = -mg\ell \cdot \sin(z)$

- torque is the rate of change of the angular momentum $\tau = \frac{dL}{dt}$
- for rotating body: $L = I \cdot \dot{z}$ "inertia tensor"

- therefore, $\tau = \frac{dL}{dt} = I \cdot \ddot{z}$ and $-mg\ell \sin(z) = I\ddot{z} \equiv m\ell\ddot{z}$

  simple for point mass ↑ ... ↑ for a point mass at the tip

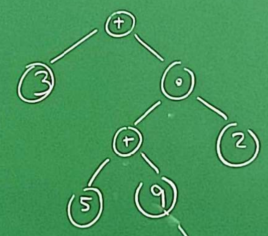
- Equation simplifies to $\ddot{z} = -\frac{g}{\ell} \sin(z)$
- For convenience, set $g = \ell = 1$ ⟹ $\boxed{\ddot{z} = -\sin(z)}$

What is x in the Pendulum system?

- cartesian coordinates $x_t \in \mathbb{R}^2$
- discretized "video" of the pendulum $x_t \in \mathbb{R}^{W \times H}$

First, stick with z: How do we find the equation $\ddot{z} = f(z, \dot{z}) = -\sin(z)$

Symbolic Regression: Methods:

$3 + (5+9) \cdot 2 \;\hat{=}$ 

- SINDy
- PySR : evolutionary algorithm on expression trees
- AI Feynman : recursively exploit symmetries, separability, compositionality of equations

Sparse Identification of Non-Linear Dynamics (SINDy) [Brunton et al. 2015]

Goal: find f s.t. $\ddot{z} = \underset{\text{RHS}}{f(z, \dot{z})}$

Motivation: Symbolic Regression scales exponentially with the depth of the expression tree

example $= F = G\,\frac{m_1 m_2}{r^2}$ "Newton's law of gravitation"

- 4 operations, 5 terms
- with binary operators $+, -, \cdot, /$ there are already $O(100)$ or $O(1000)$ possible rearrangements of the formula

$\rightarrow$ brute force is not an option

idea: "flatten the expression tree"

Realization via Feature-Augmented Lasso Regression $\quad \underbrace{\approx O(10)}$

- simplify and limit the RHS to a linear combination of (non-) linear terms

- augment the data z with polynomials, time derivative of z, custom terms
- based on prior knowledge

Example $\ddot{z} = \xi_1 \cdot 1 + \xi_2 \cdot z + \dots + \xi_L \cdot \dot{z}^2 \sin(z)$

Rewrite as a coefficient matrix Ξ and a term "library" matrix θ

$\Xi \in \mathbb{R}^{L \times D}$ with L terms, $D = \dim(z)$ initialized with ones o Gaussian values

$\theta(z, \dot{z}) \in \mathbb{R}^{T \times L}$ with L terms, T snapshots of the system $(z, \dot{z}, \ddot{z})_t$

augmented training data

at each time step ; $\ddot{z}_t = \theta(z, \dot{z})_t \, \Xi$

→ Regression problem! We know how to solve these!

Solve for Ξ, $\hat{\Xi} = \underset{\Xi}{\operatorname{argmin}} \sum_{t=1}^{T} \| \ddot{z}_t - \theta(z, \dot{z})_t \, \Xi \|_2^2$

Problems: 1. Many higher order terms overfit the data

2. not sparse ←

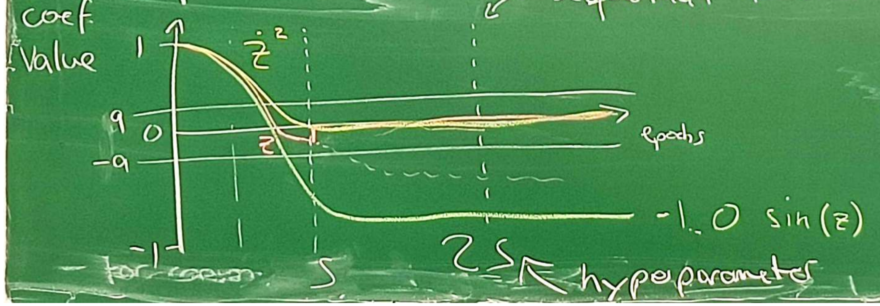
Two methods to encourage sparsity :

- L_1 regularization $\rightarrow$ $\hat{\Xi} = \arg\min_{\Xi} \sum_{t=1}^{T} \| \ddot{z}_t - \Theta(z,\dot{z})_t \, \Xi \|_2^2 + \lambda_1 \sum_{i=1} \sum_{d=1} | \Xi_{id} |$

- Thresholding "clean up"
    - Sequential Thresholding : Every S epochs during training , turn off coefficients that are very small (i.e. set to 0 and exclude from further training)

- Careful Thresholding Methods : setting coef. to 0 after they show convergence

Example.

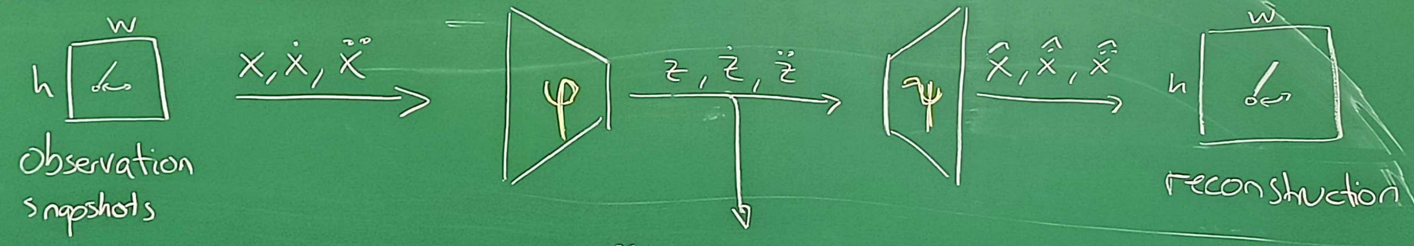


- Most of the coefficients "converge" to 0 and do not contribute to the equation

$\Xi_{\sin(z)} = -1$ ideal solution

$\Xi_{other} = 0$

SINDy – Autoencoder



$$\ddot{z} = \Theta(z, \dot{z}) \, \boxed{}$$

learn dynamics equation

learn jointly!

- Encoder φ and decoder ψ : fully-connected Why? easy chain rule
- for pendulum : $d(z) = 1$, in general : hyperparameter

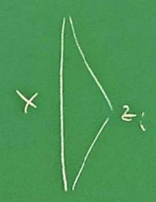
How to obtain $\dot{x}$, $\ddot{x}$

- direct measurement
- numerical differentiation $\dot{x}(t) \approx \frac{x(t) - x(t - \Delta t)}{\Delta t}$
- if simulated in z : via chain rule of the artificial embedding

Propagation of Time Derivatives

Fist time drivative

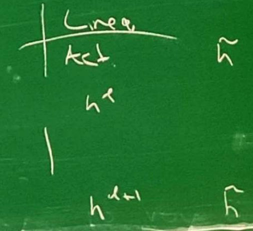
$$\frac{dz_i(x(t))}{dt} = \sum_j \frac{dz_i}{dx_j}\,\frac{dx_j}{dt}$$



Realization with fc-network

- Start with $\tilde{h}^0 = xW^0 + b^0$
- Hidden layers: $\tilde{h}^l = g(\tilde{h}^{l-1})W^l + b^l$ $\quad z = \tilde{h}^L$
- Propagate the drivative through each layer

$$\frac{d\tilde{h}^l}{dt} = \left(g'(\tilde{h}^{l-1}) \odot \frac{d\tilde{h}^{l-1}}{dt} \right) W^l$$



Second time derivative

$$\ddot{z}_i = \sum_{j,k} \dot{x}_j \dot{x}_k \frac{\partial^2 z_i}{\partial x_j \partial x_k} + (\nabla_x z_i)\ddot{x}$$

For linear layers: $(\tilde{h}^l)_i = (h^{l-1} W^l + b^l)_i = \sum_j h_j^{l-1} W_{ij}^l + b^l$

$$\rightarrow \frac{\partial \tilde{h}_i^l}{\partial h_j^{l-1}} = W_{ij}^l$$ does not depend on h_j^{l-1} anymore

$\Rightarrow$ first term vanishes

For point-wise activation functions, first term simplifies

first term $= \dot{\tilde{h}}^l \odot \dot{\tilde{h}}^l \odot g''(\tilde{h}^l)$ since $i=j=k$

Algorithm ; Transformation of Time Derivatives

Given $x, x, \ddot{x}, g, g', g''$ and W^{ℓ}, b^{ℓ}

① Initalize $z^{(0)}, \dot{z}^{(0)}, \ddot{z}^{(0)} \leftarrow x, \dot{x}, \ddot{x}$

② For $\ell = 1$ to $\ell = L$

ⓐ $\tilde{z}^{\ell} \leftarrow z^{\ell-1} W^{\ell} + b^{\ell}$

$z^{\ell} \leftarrow g(\tilde{z}^{\ell})$

ⓑ $\dot{\tilde{z}}^{\ell} \leftarrow \dot{z}^{\ell-1} W^{\ell}$

$\dot{z}^{\ell} \leftarrow g'(\tilde{z}^{\ell}) \odot \dot{\tilde{z}}^{\ell}$

ⓒ $\ddot{\tilde{z}}^{\ell} \leftarrow \ddot{z}^{\ell-1} W^{\ell}$

$\ddot{z}^{\ell} \leftarrow (g''(\tilde{z}^{\ell}) \odot \dot{\tilde{z}}^{\ell}) \odot \dot{\tilde{z}}^{\ell} + (g'(\tilde{z}^{\ell}) \odot \ddot{\tilde{z}}^{\ell})$

Outputs $z^{L}, \dot{z}^{L}, \ddot{z}^{L}$