Exam

Here you can schedule your exam and upload your project notebook

Lecture Notes

Lecture notes will be added weekly.

Dear students,
in order to take the exam you must first schedule an appointment by using the **Exam Dates** module below. Please be aware that booking a date is binding!
You can book an appointment up to 8 workdays prior to the chosen date (e.g. exam date: Thursday, 11.03. --> booking deadline: Monday, 01.03 - 00:00) .
Timeslots/dates may be added when the given ones are taken.
You can withdraw from the exam only by email (to cihan.ates@kit.edu, katharina.stichling@kit.edu and dogan.bicat@kit.edu)!

After booking the exam date, you need to upload your complete project notebook to the **Project Upload Folder** no later than 4 workdays previous to your exam date.
Only with this notebook you will be admitted to the exam!
The booking of an exam date and uploading of the notebook can be done at different times, provided that the time limit is met.

CONTENT

Exam Dates

Here you can book an appointment for the oral exam

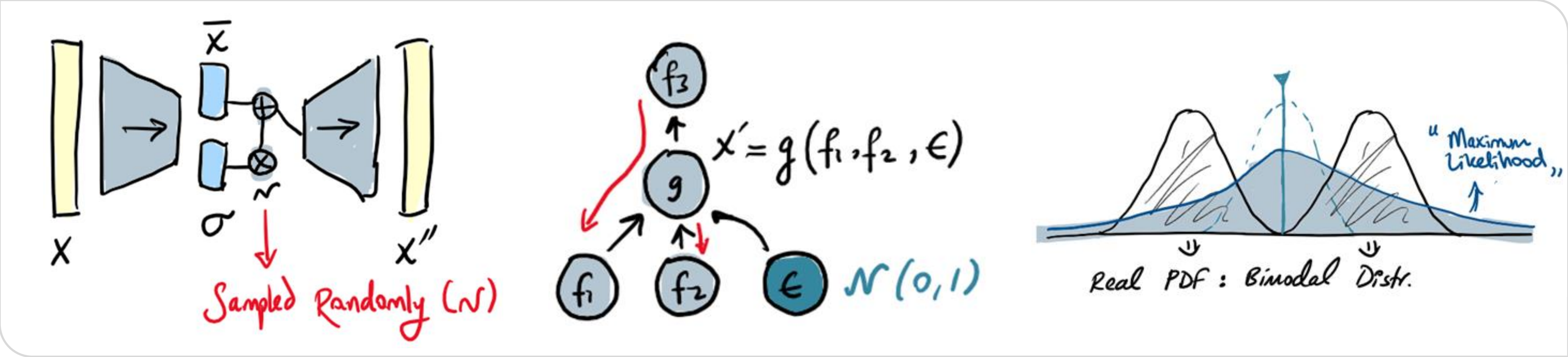
Project Upload Folder

Here you can upload your project notebook. Uploading is mandatory for the exam

Data Driven Engineering I: Machine Learning for Dynamical Systems

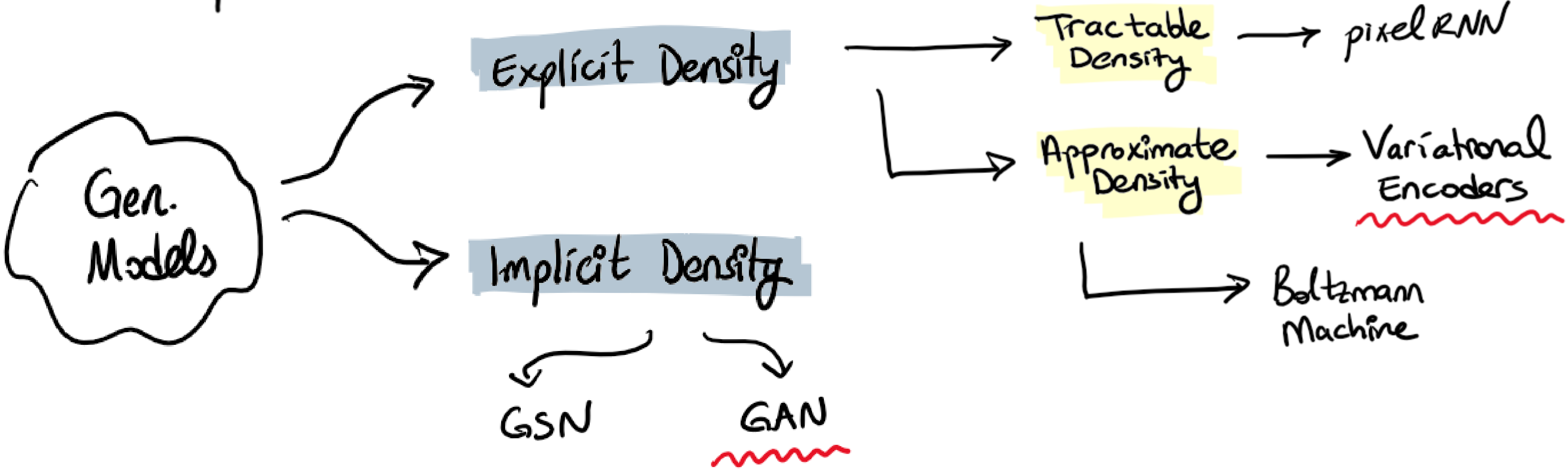
Introduction to Generative Learning: VAEs and GANs

Institute of Thermal Turbomachinery
Prof. Dr.-Ing. Hans-Jörg Bauer



UL → Generative Models :

* It is probabilistic in nature



Generative Adversarial Networks : GANs

- Generative ⇒ creating non-existing data
- Adversarial ⇒ Competitive dynamics (game-like)
- Network ⇒ Neural networks
- GANS (2014) → Generator, → Discriminator
- Random Numbers → $\boxed{\text{Generator}}$ → Fake Examples

  Obj: Be as realistic as possible
- Real examples → $\boxed{\text{Discriminator}}$ → probability ⇒ [0, 1]

  Fake Examples → $\boxed{\text{Discriminator}}$

  Obj: Distinguish Fake from Real data

Training Algorithm:

For each training do:

Train (D):

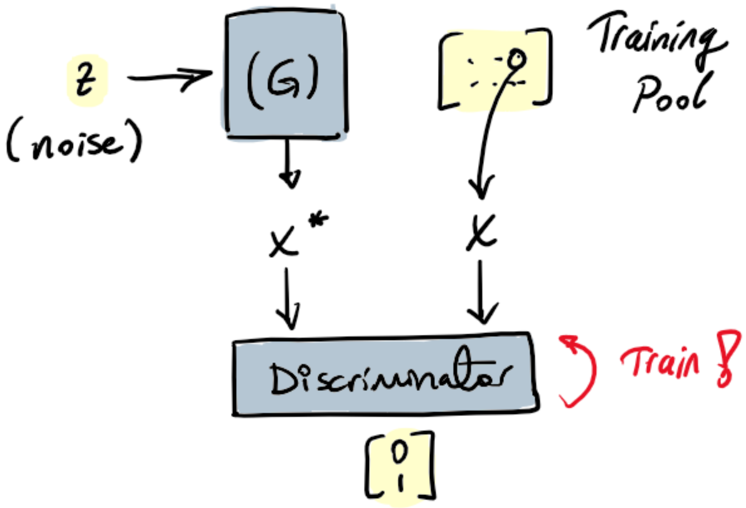
(1) Take a random real example from Training data, x

(2) Get a fake example from Generator, x^*

(3) Use Discriminator to classify x & x^*.

(4) Compute the class. error.

(5) Backprop. error & update Discriminator trainable parameters.



Training Algorithm:

Train (G):

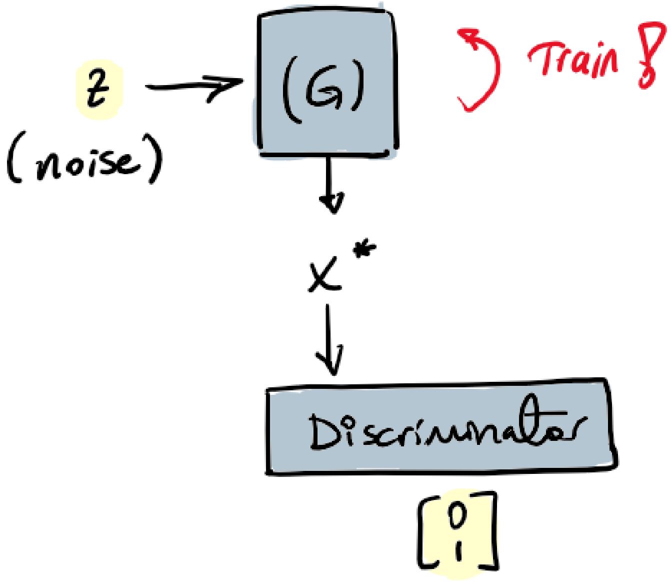
(6) Generate a new fake x^*.

(7) Use Discriminator to classify x^*.

(8). Compute the error.

(9). Update Generator's trainable parameters via backprop.

end for



Training GANs

* In MLP, we have a clear goal & measure

  Eg Minimize Cross-entropy loss.

* In GANs, two networks have competing obj. !

  (G) ↑ ; (D) ↓ // (D) ↑ ; (G) ↓

Training GANs

Nash Equilibrium := Point where neither "player" can improve their situation

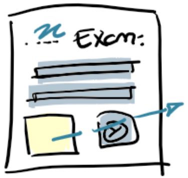
- (G) := Fakes are indistinguishable from Real data
- (D) := at best randomly guess (F/R ⇒ 1)

In practice ; ~ impossible to achieve Nash Eq.

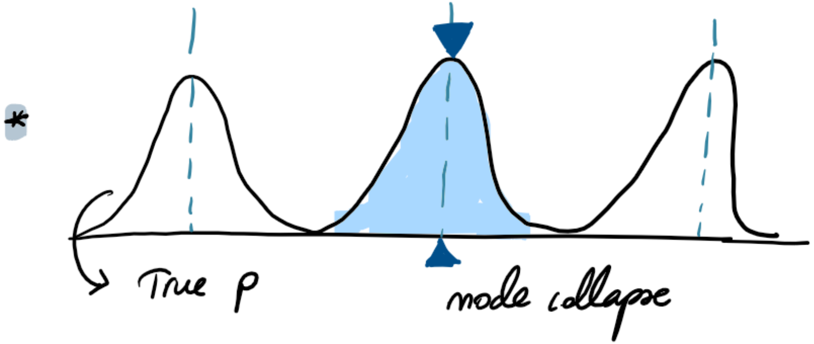
it still works ...

Training GANs

Training problems #1: Mode Collapse



"Abracadabra ?" ~ "I create as I speak ?"



$$\begin{bmatrix} -4 \\ -2 \\ 0 \\ +2 \\ +4 \end{bmatrix}$$

True values

$\boxed{G} \Rightarrow$ "0" (✓)

~~[-4, -2, 2, 4]~~

Training GANs

Training Problem #2 : Over generalization

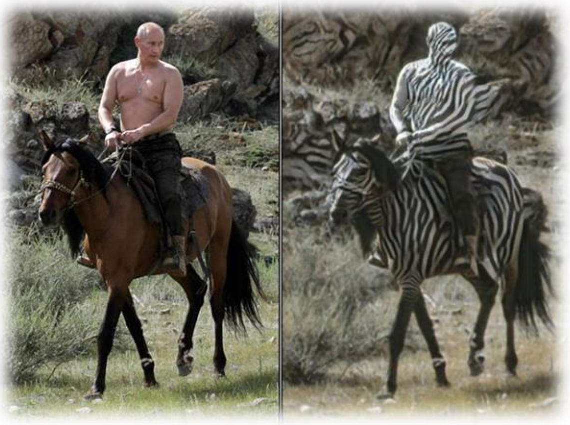
* Mods that should not exist, do exist.

* $\begin{bmatrix}-4\\-2\\0\\+2\\+4\end{bmatrix} \quad \boxed{G} \begin{array}{l}\nearrow -2/3\\ \searrow 1/2\\ \quad \ldots\end{array}$

True values [intigers]

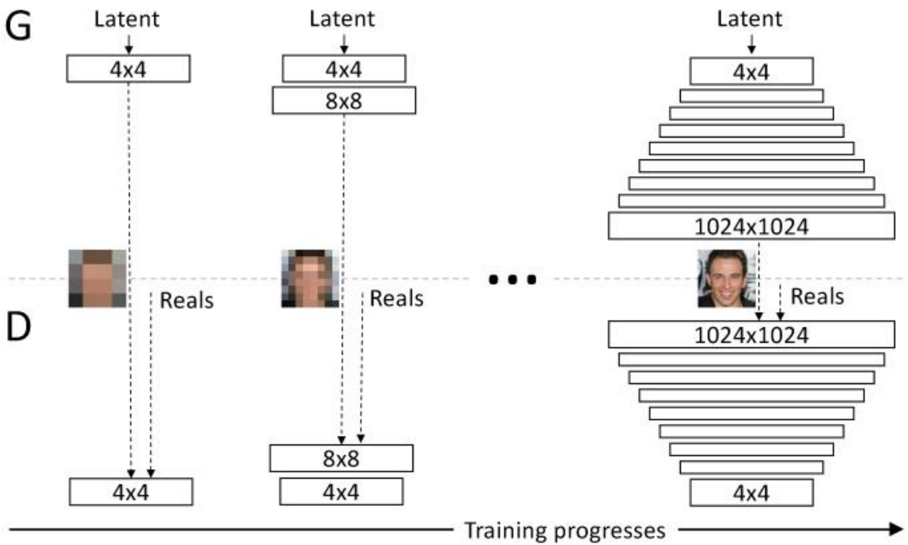
[real numbers]

* Image generation



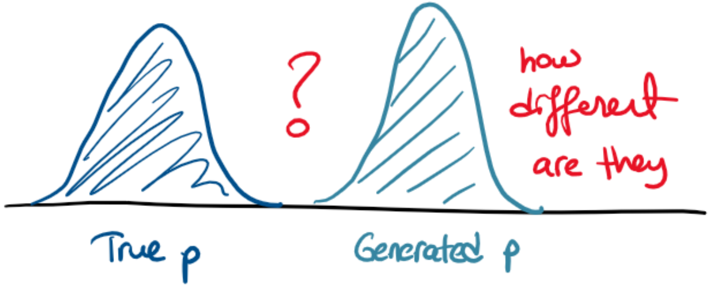
Possible Remedies

① Growing the network gradually

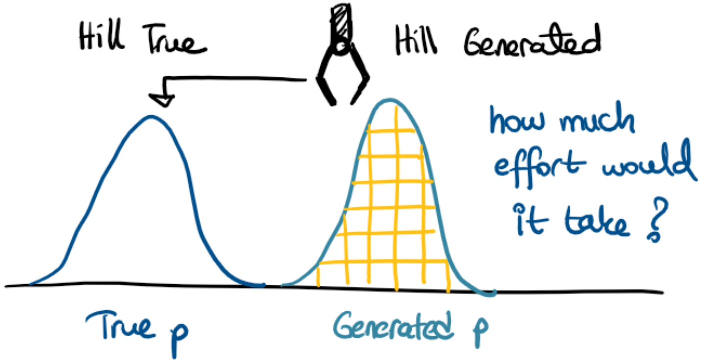


Possible Remedies

② Alternative loss definitions ⇒ Wasserstein GAN



* Distance := (TV distance, KL divergence, JS divergence, ...) ⇒ Earth-mover Distance
(Similarity)



Possible Remedies

② Alternative loss definitions ⇒ Wasserstein GAN

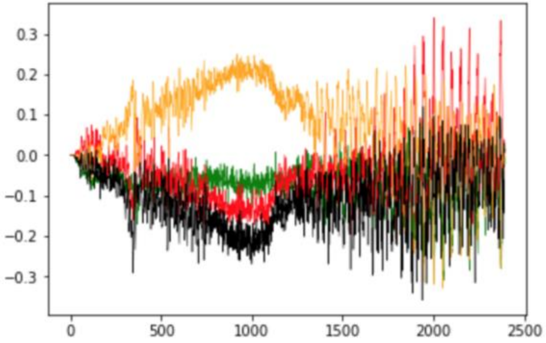
* Class. ⇨ [0, 1]
  ↳ Train... ⇨

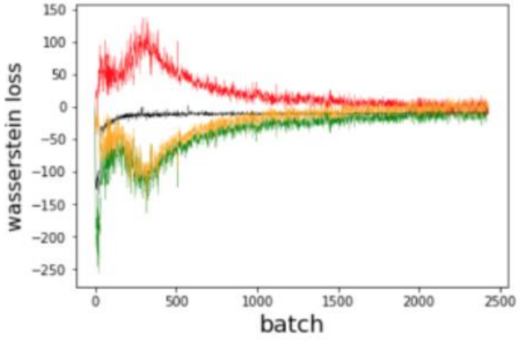
0.9971	0.0004
0.9965	0.00013
0.9981	0.00021
...	

⇓

vanishing gradient problem

* Stabilized learning

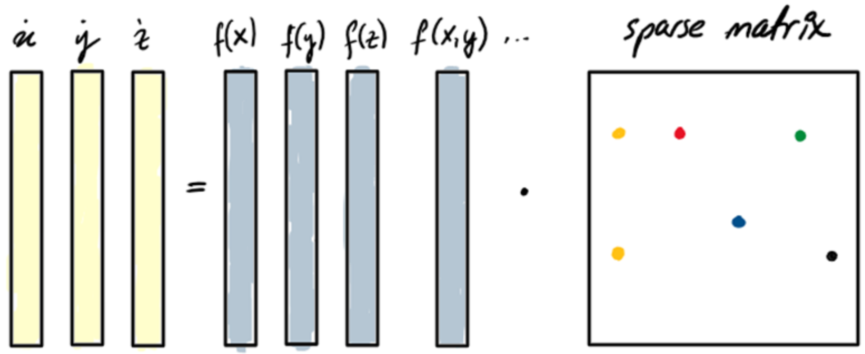


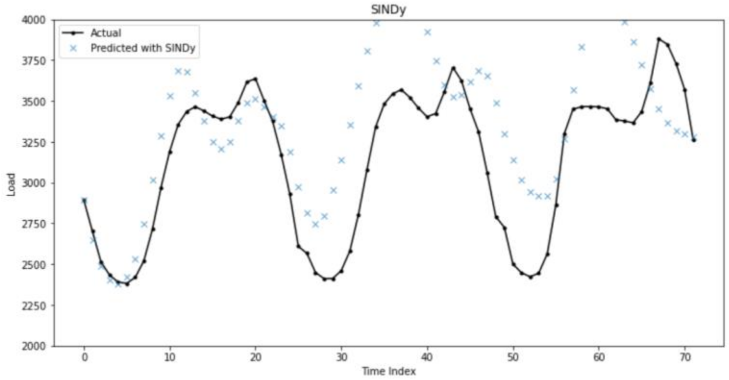


Data Driven Engineering I: Machine Learning for Dynamical Systems

Introduction to Data Driven Control
Data Driven Discovery & Linear System Identification

Institute of Thermal Turbomachinery
Prof. Dr.-Ing. Hans-Jörg Bauer





DDE: Dynamical Systems

Discovery // Characterization // Simulation

Understanding how ... works

* "Science" := $\underbrace{\text{interpret. of observations}}$ ... in a systematic way
* Prereq. ⇒ organized "book keeping"
* $\underbrace{\text{Scientific Data} \Rightarrow \text{Discovery} \Rightarrow \text{Management Optimization}}$ "Engineering" } interpret. as governing model eqns.

$$f(\underset{\text{state}}{x}, \overset{\text{Transient}}{t}, \underbrace{\alpha, \beta, \gamma \ldots}_{\text{parameters}})$$

DDE: Dynamical Systems

Discovery // Characterization // Simulation

Understanding how ... works

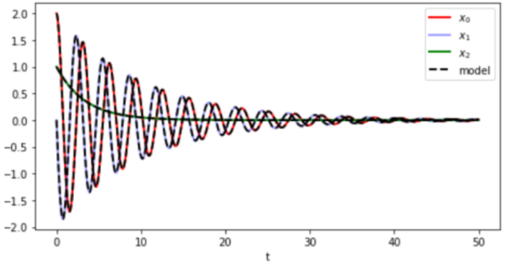
*

$$f(\underbrace{x}_{\text{state}}, \overset{\text{Transient}}{t}, \underbrace{\alpha, \beta, \gamma \ldots}_{\text{parameters}}) \xrightarrow[\text{observation}]{\text{Modeling}}$$

→ Traditional approach → First Principles

- - → "Theory of Everything"

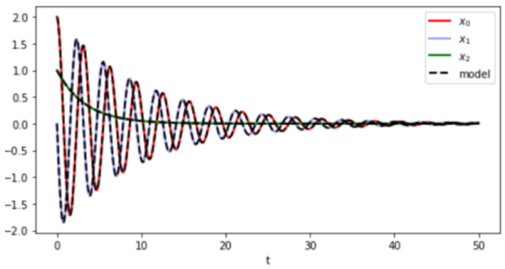
→ Data-Driven Model Discovery



DDE: Dynamical Systems

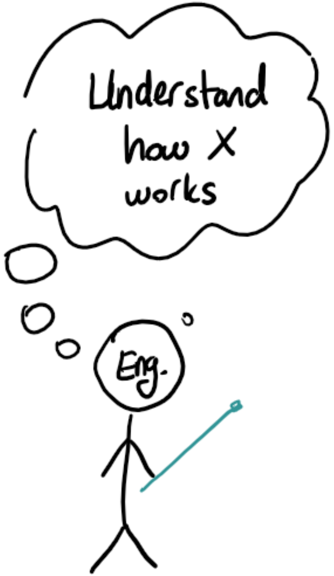
Discovery // Characterization // Simulation

Understanding how ... works



Data-Driven Model Discovery

$\underbrace{f(x,t,\overset{\nearrow \text{Transient}}{\alpha,\beta,\gamma\ldots})}_{\text{parameters}}$, x: state



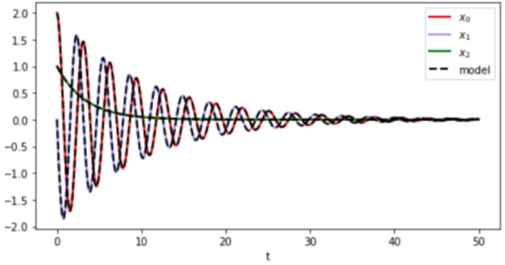
this 'model' would let me...

- Interpretability: What if..
- Design & Optimization
- Future state prediction
- Active control with feedback

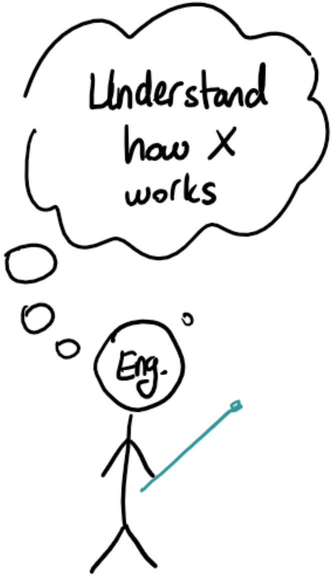
DDE: Dynamical Systems

Discovery // Characterization // Simulation

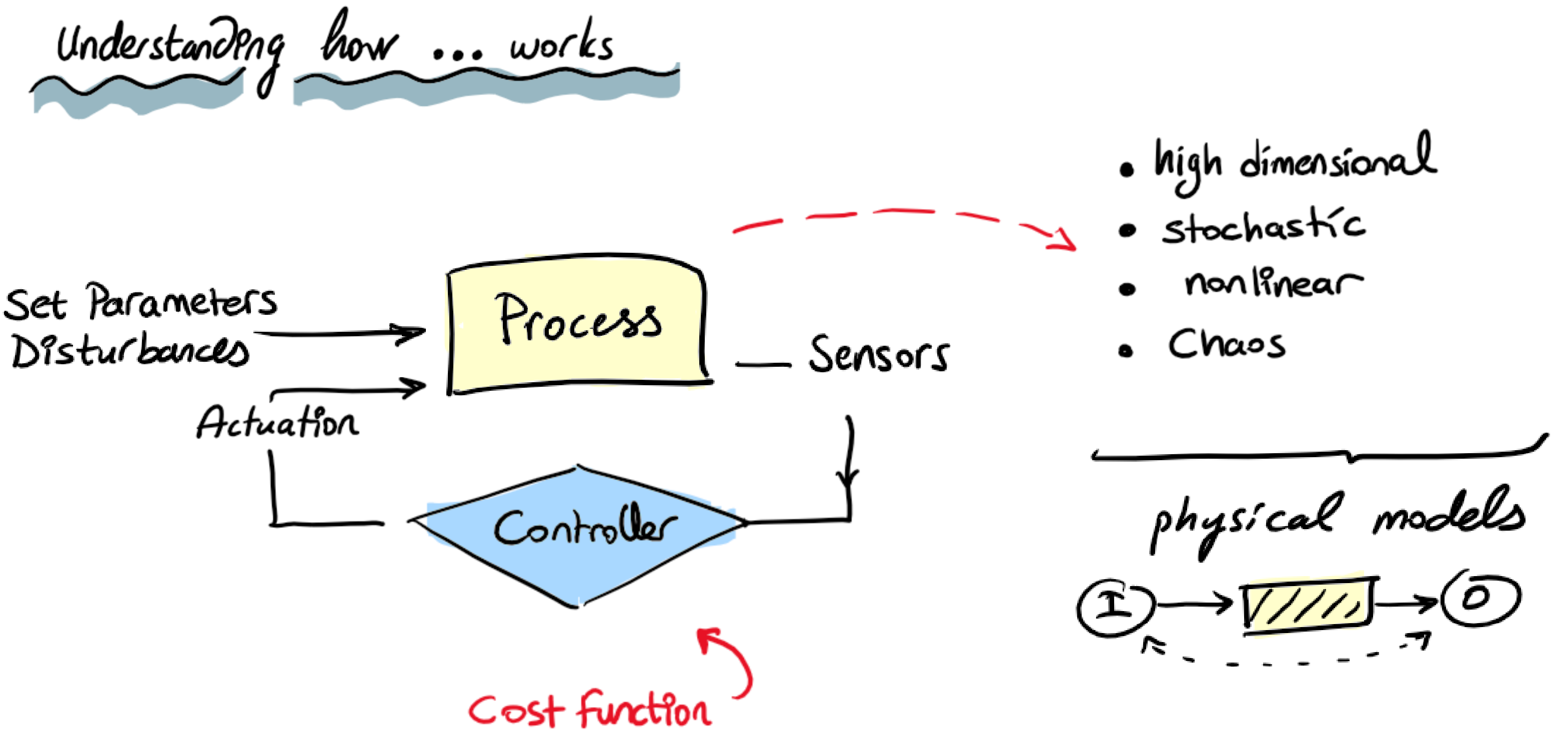
Understanding how ... works



Data-Driven Model Discovery

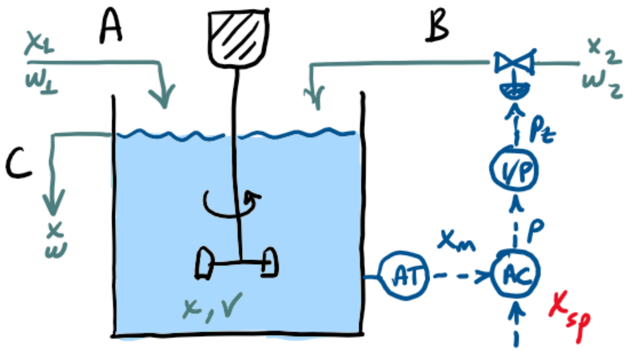


DDE: Dynamical Systems
Discovery // Characterization // Simulation



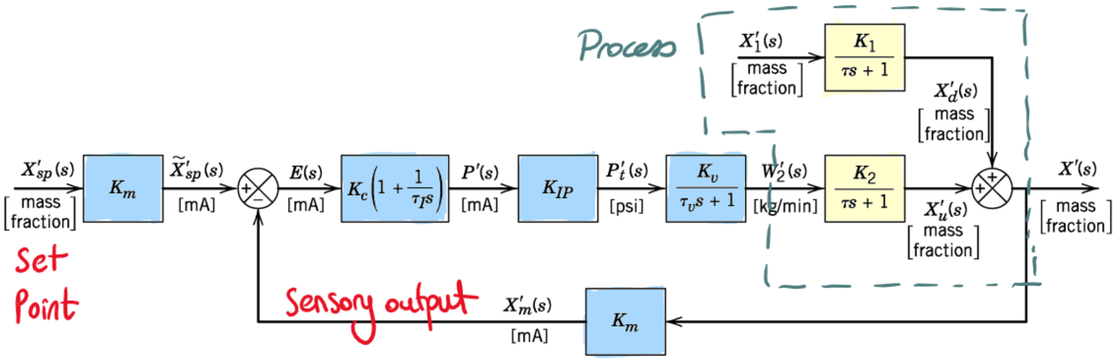
DDE: Dynamical Systems
Discovery // Characterization // Simulation

* FACT : Process ⇔ Controller
linked



Stirred Tank

Closed-loop Control System :



DDE: Dynamical Systems

Discovery // Characterization // Simulation

Karlsruhe Institute of Technology

* FACT : Process ⇔ Controller

linked &

Tasks:

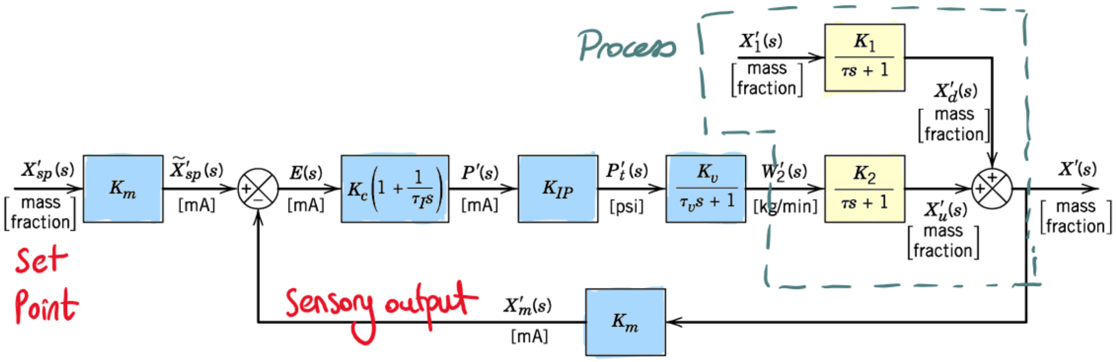
① Create a phy. model | DDE-I

② Create a controller model

DDE-II

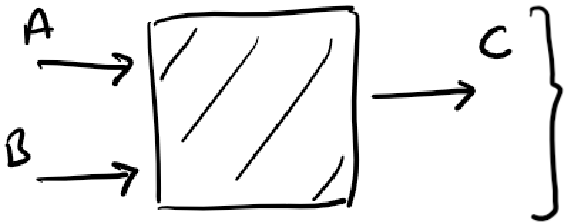
③ Coupled optimization

Closed-loop Control System:



DDE: Dynamical Systems
Discovery // Characterization // Simulation

Understanding how ... works



Objective :

How to find a physical ~~model~~ from data

manageable model !

⇒ Regression

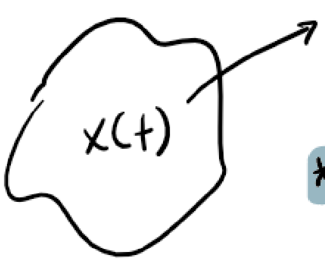
Sparse regression

...

symbolic regression

How SINDy works?

SINDy: Sparse Identification of Non-linear Dynamics



Collected data

system

* Assume: system can be modelled as a dynamical system

$\Rightarrow$ $$\frac{dx}{dt} = f(x(t))$$

$\downarrow$

Objective

* $$\begin{pmatrix} x \\ x' \end{pmatrix} \rightarrow \boxed{\text{SINDy}} \rightarrow f$$

How SINDy works?

① $\frac{d}{dt} X = X' = f(X)$ } Regression Problem

↳ state (measurements) ⇒ matrix

$X =$ 

states @ $t = t_i$

@ $t = t_{i+1}$

@ $t = t_{i+2}$

$$\Rightarrow X = \begin{bmatrix} x_1(t_1) & x_2(t_1) & \cdots & x_n(t_1) \\ x_1(t_2) & x_2(t_2) & \cdots & x_n(t_2) \\ \vdots & \vdots & & \vdots \\ x_1(t_m) & x_2(t_m) & & x_n(t_m) \end{bmatrix}$$

$$\Rightarrow X' = \begin{bmatrix} x_1'(t_1) & x_2'(t_2) & \cdots & x_n'(t_1) \\ x_1'(t_2) & x_2'(t_2) & \cdots & x_n'(t_2) \\ \vdots & \vdots & & \vdots \\ x_1'(t_m) & x_2'(t_m) & \cdots & x_n'(t_m) \end{bmatrix}$$

How SINDy works?

① $\frac{d}{dt}X = X' = f(X)$

② Construct hypothesis library Θ

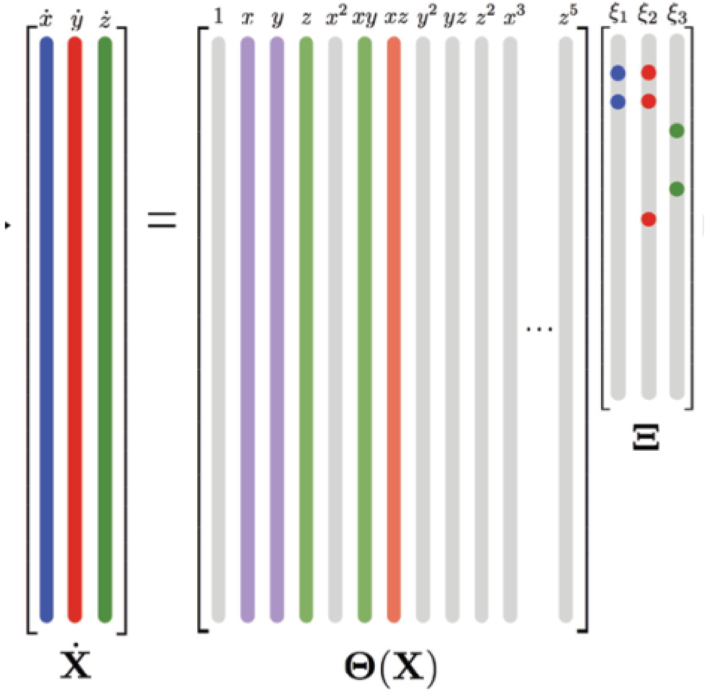
$$f(X) \approx \Theta(X)\Xi$$

sparse matrix

Matrix:

$$\begin{bmatrix} | & | & & | \\ \theta_1 & \theta_2 & \cdots & \theta_\ell \\ | & | & & | \end{bmatrix} \qquad \theta(x) = \begin{bmatrix} | & | & & | \\ x_1(t)^2 & x_1(t)x_2(t) & \cdots & x_n^2(t) \\ | & | & & | \end{bmatrix}$$

$$\Xi = \begin{bmatrix} | & | & | & & | \\ \xi_1 & \xi_2 & \xi_3 & \cdots & \xi_n \\ | & | & | & & | \end{bmatrix}$$



How SINDy works?

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow f = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$$

* $\dfrac{dx}{dt} = f(x) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}$

* $f_1(x) = \sum_i^{\infty} \sum_j^{\infty} a_{ij} \, x_1^{\,i} x_2^{\,j}$

* $f_2 = \sum_i^{\infty} \sum_j^{\infty} b_{ij} \, x_1^{\,i} x_2^{\,j}$

* $\dfrac{dx}{dt} = f(x) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} 1 - x_1 + x_1 x_2 \\ x_1^2 - 2x_2^2 \end{bmatrix}$

Loss function:

* Column k,

$$\mathcal{L} \Rightarrow \underbrace{\| X'_k - \Theta(X)\xi_k \|_2}_{\text{"Reconstruction Loss"}} + \lambda \underbrace{\| \xi_k \|_1}_{\text{"Sparsity Cost"}}$$

How SINDy works?

① $\frac{d}{dt}x = x' = f(x)$

What do we need?

* $[x]$ → must have
* $[x']$ → have it / calculate numerically
* Hypothesis Library θ

PySINDy

What is available in PySINDy ?

$$X' \approx \Theta(X)\,\Xi$$

pysindy.differentiation

$\Downarrow$

computes X' from X

pysindy.feature_library

$\Downarrow$

$\Theta(X)$ alternatives

pysindy.optimizers

$\Downarrow$

solve regression problem $\Rightarrow \Xi$

colab

Additional Notes

Karlsruhe Institute of Technology

Exercise Design

1. Very basic implementation
2. Sci-kit learn → Grid Search Option
3. How to use Data Frames.
4. Case Study → univariate load prediction