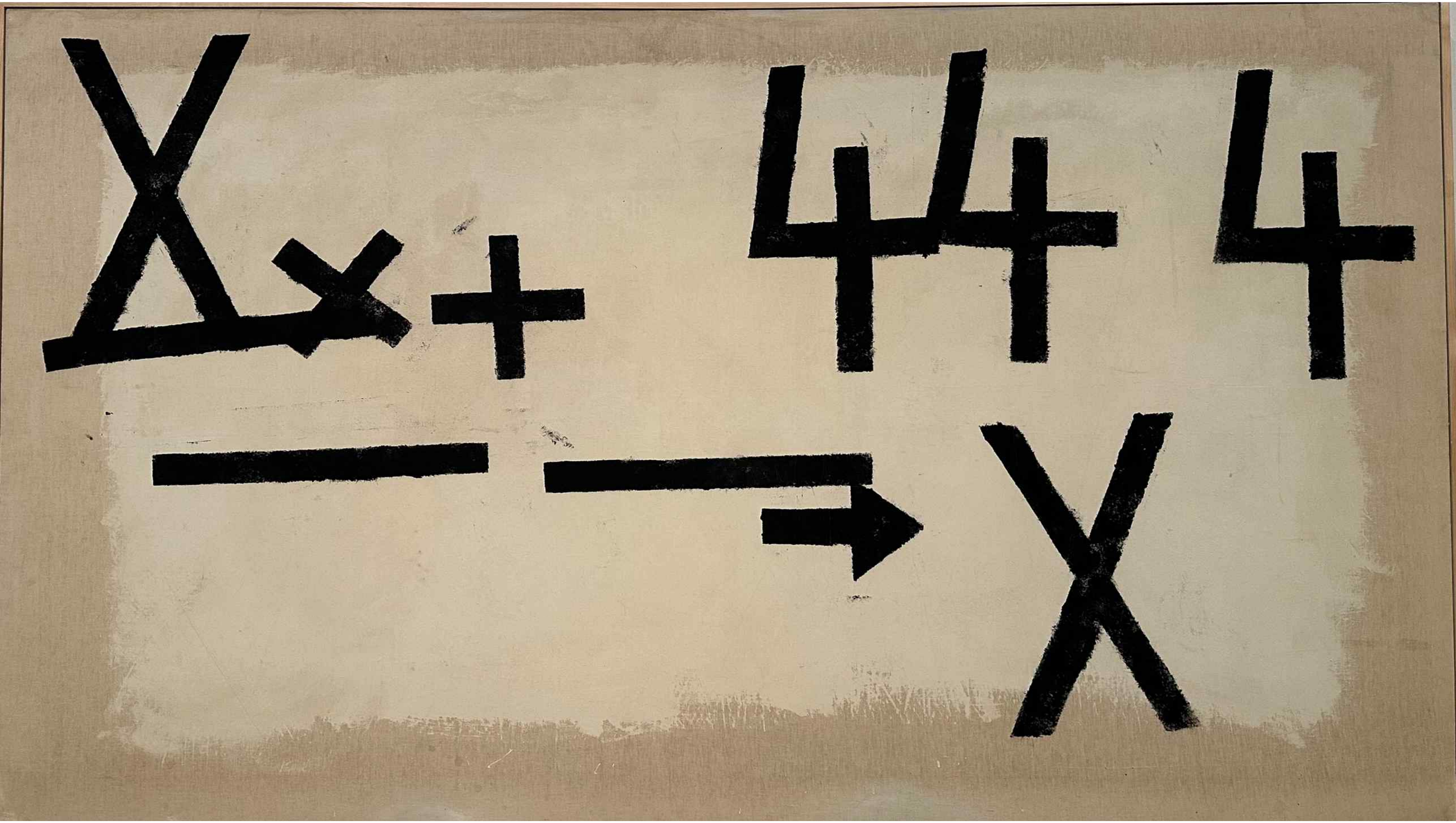
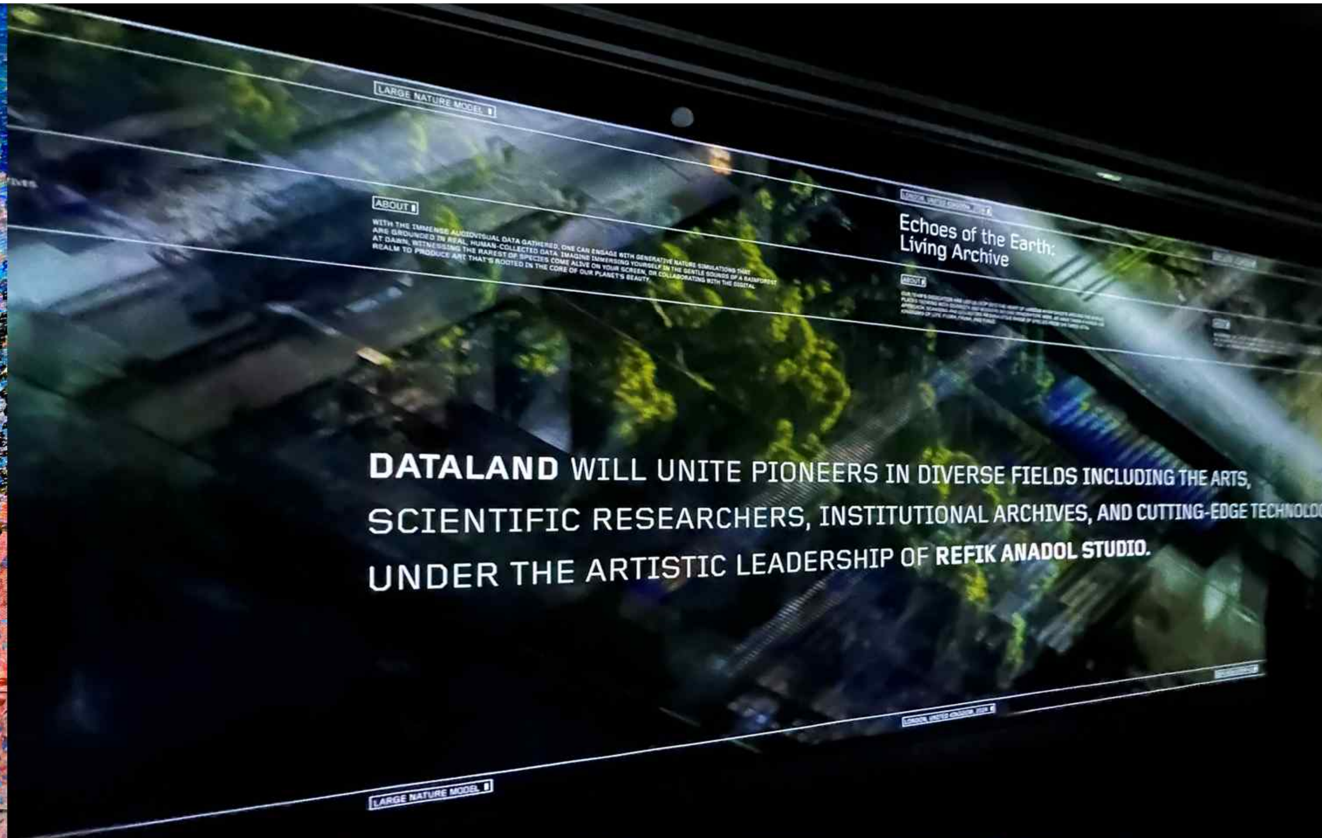
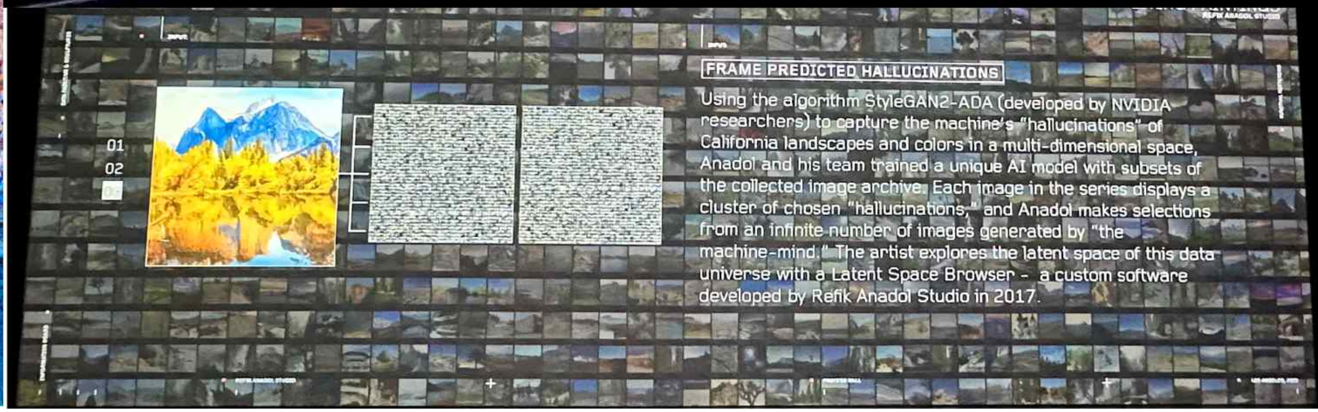
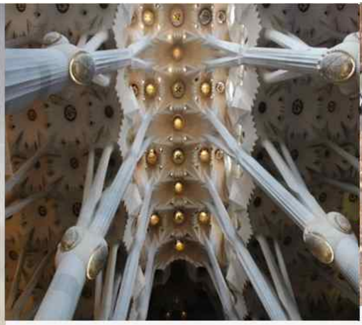
Understanding and Applications of AI from Engineering Perspective

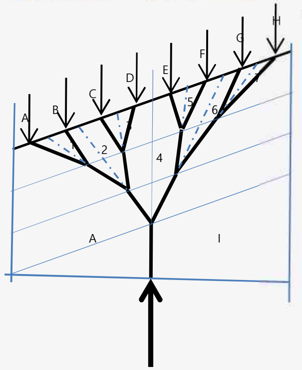
SNU
2024.04.04, 4PM
홍성걸

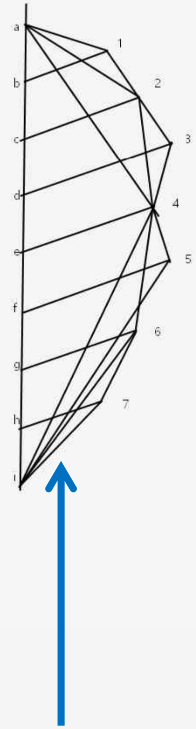



LARGE NATURE MODEL
ABOUT
Echoes of the Earth:
Living Archive
DATALAND WILL UNITE PIONEERS IN DIVERSE FIELDS INCLUDING THE ARTS,
SCIENTIFIC RESEARCHERS, INSTITUTIONAL ARCHIVES, AND CUTTING-EDGE TECHNOLOG
UNDER THE ARTISTIC LEADERSHIP OF REFIK ANADOL STUDIO.
LARGE NATURE MODEL


01
02
FRAME PREDICTED HALLUCINATIONS
Using the algorithm StyleGAN2-ADA (developed by NVIDIA researchers) to capture the machine's "hallucinations" of California landscapes and colors in a multi-dimensional space, Anadol and his team trained a unique AI model with subsets of the collected image archive. Each image in the series displays a cluster of chosen "hallucinations," and Anadol makes selections from an infinite number of images generated by "the machine-mind." The artist explores the latent space of this data universe with a Latent Space Browser - a custom software developed by Refik Anadol Studio in 2017.





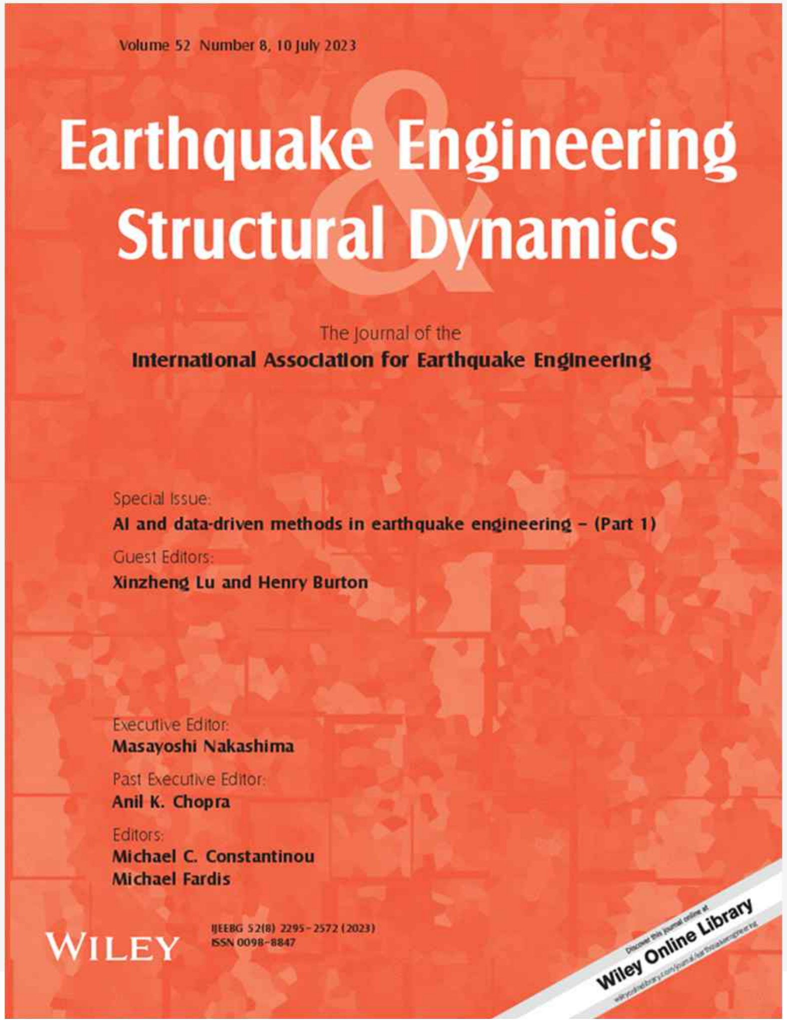


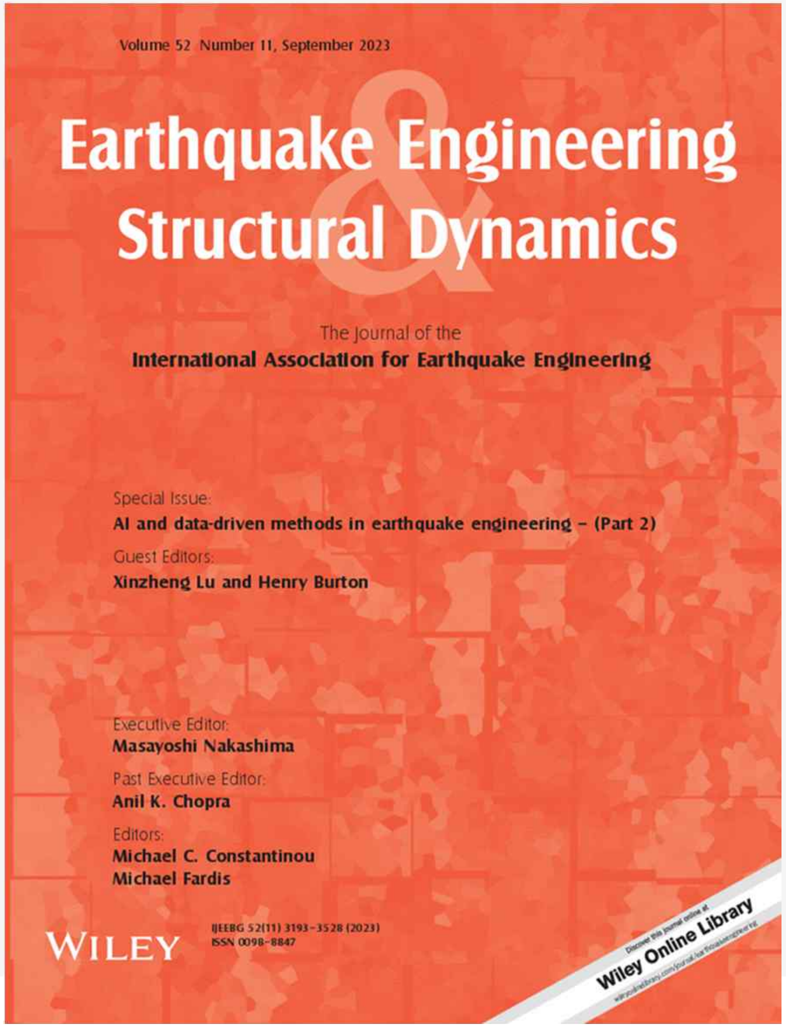
홍성걸 sglhong@snu.ac.kr

1996년 이후 서울대 건축학과 교수
전 서울대 중앙도서관 관장
전 지진공학회 회장
전 콘크리트학회 부회장

건축구조시스템과 구조설계, 전통 목조 및 석탑, Graphic statics,
콘크리트재료에 관심이 많으며
첨단건설재료(UHPC) 개발을 통해 3-D 프린팅으로 구조시스템
개발 연구 그리고 AI applications와 Quantum computing

AI Application to Earthquake Engineering





SPECIAL ISSUE ARTICLES

SPECIAL ISSUE ARTICLES

Full Access

Unsupervised machine learning for detecting soil layer boundaries from cone penetration test data

Kenneth S. Hudson, Kristin J. Ulmer, Paolo Zimmaro, Steven L. Kramer, Jonathan P. Stewart, Scott J. Brandenberg

A continuous Bayesian network regression model for estimating seismic liquefaction-induced settlement of the free-field ground

Jilei Hu, Bin Xiong, Zheng Zhang, Jing Wang

Machine learning-based prediction of the seismic response of fault-crossing natural gas pipelines

Wenyang Zhang, Francois Ayello, Doug Honegger, Yousef Bozorgnia, Ertugrul Taciroglu

Machine-learning-based optimization framework to support recovery-based design

Omar Issa, Rodrigo Silva-Lopez, Jack W. Baker, Henry V. Burton

Base-isolation design of shear wall structures using physics-rule-co-guided self-supervised generative adversarial networks

Wenjie Liao, Xinyu Wang, Yifan Fei, Yuli Huang, Linlin Xie, Xinzheng Lu

A deep learning method to monitor axial pressure and shear deformation of rubber bearings under coupled compression and shear loading

Collaborative filtering-based collapse fragility assessment

Xingquan Guan, Henry V. Burton

Uncertainty-aware structural damage warning system using deep variational composite neural networks

Kareem A. Eltouny, Xiao Liang

Multi-channel response reconstruction using transformer based generative adversarial network

Wenhao Zheng, Jun Li, Qilin Li, Hong Hao

Geometry-guided semantic segmentation for post-earthquake buildings using optical remote sensing images

Yu Wang, Xin Jing, Yang Xu, Liangyi Cui, Qiangqiang Zhang, Hui Li

Deep neural network-based regional seismic loss assessment considering correlation between EDP residuals of building structures

Chulyoung Kang, Taeyong Kim, Oh-Sung Kwon, Junho Song

Efficient regional seismic risk assessment via deep generative learning of surrogate models

Shanwu Li, Charles Farrar, Yongchao Yang

Simulation-based methodology to identify damage indicators and safety thresholds for post-earthquake evaluation of structures

Francisco A. Galvis, Anne M. Hulsey, Jack W. Baker, Gregory G. Deierlein

Generalized stacked LSTM for the seismic damage evaluation of ductile reinforced concrete buildings

Bilal Ahmed, Sujith Mangalathu, Jong-Su Jeon

Seismic damage prediction of RC buildings using machine learning

Sanjeev Bhatta, Ji Dang

Review paper on EQ engineering

remote sensing

Review paper

Deep Learning for Earthquake Disaster Assessment: Objects, Data, Models, Stages, Challenges, and Opportunities

Jing Jia andWenjie Ye*

Earthquake Disaster Assessment Framework

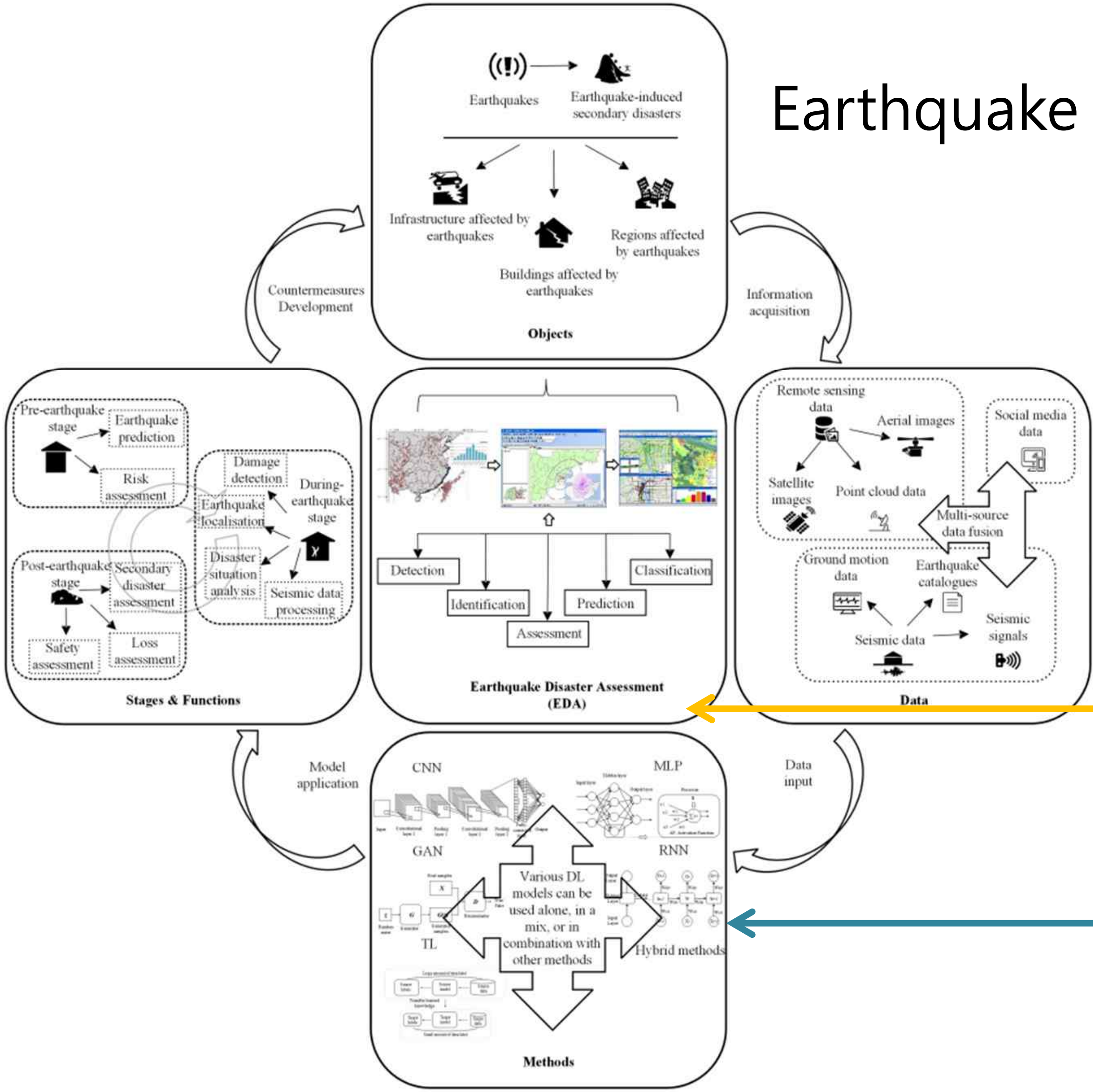


Figure 6. The application framework of DL for EDA from four dimensions.

Collection of Data

Objectives of ML and DL

Typical DL methods

Structural Health Monitoring

Large-scale structural health monitoring using composite recurrent neural networks and grid environments
By Kareem A. Eltouny Xiao Liang

COMPUTER-AIDED CIVIL AND INFRASTRUCTURE ENGINEERING

The framework relies on a 5D, time dependent grid environment and a novel spatiotemporal composite **autoencoder** network. This network is a hybrid of **autoencoder convolutional neural networks** and **long short-term memory networks**. A 10-story, 10-bay, numerical structure is used to evaluate the proposed framework damage diagnosis capabilities. The framework was successful in diagnosing the structure health state with average accuracies of 93% and 85% for damage detection and localization, respectively.

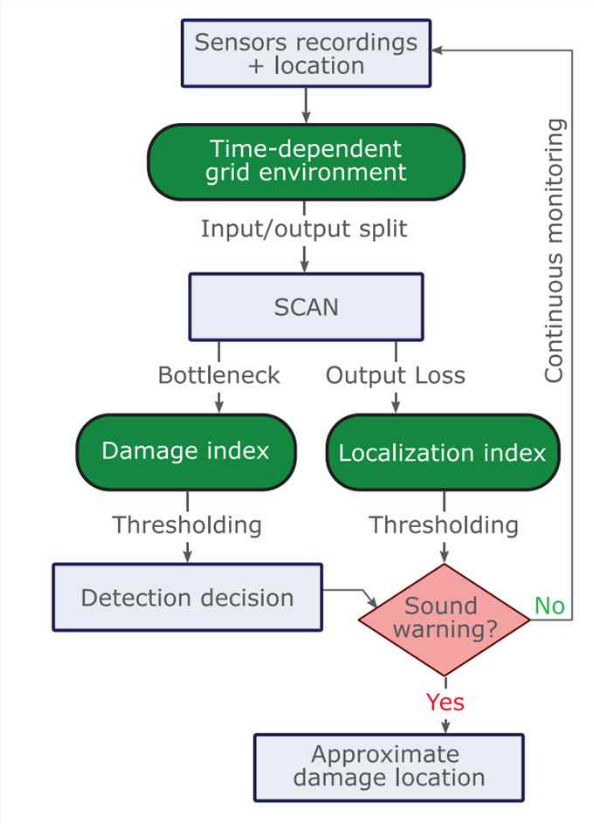


FIGURE 1 The damage diagnosis system summary

Composite autoencoder architecture

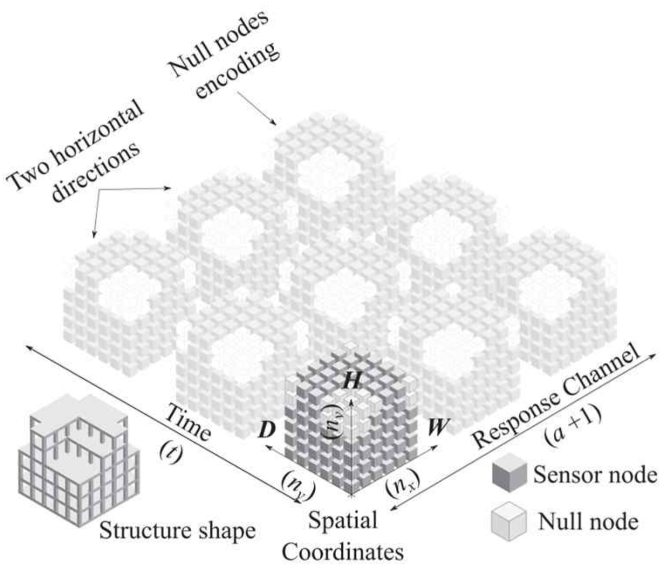


FIGURE 2 An example of the proposed time-dependent grid environment

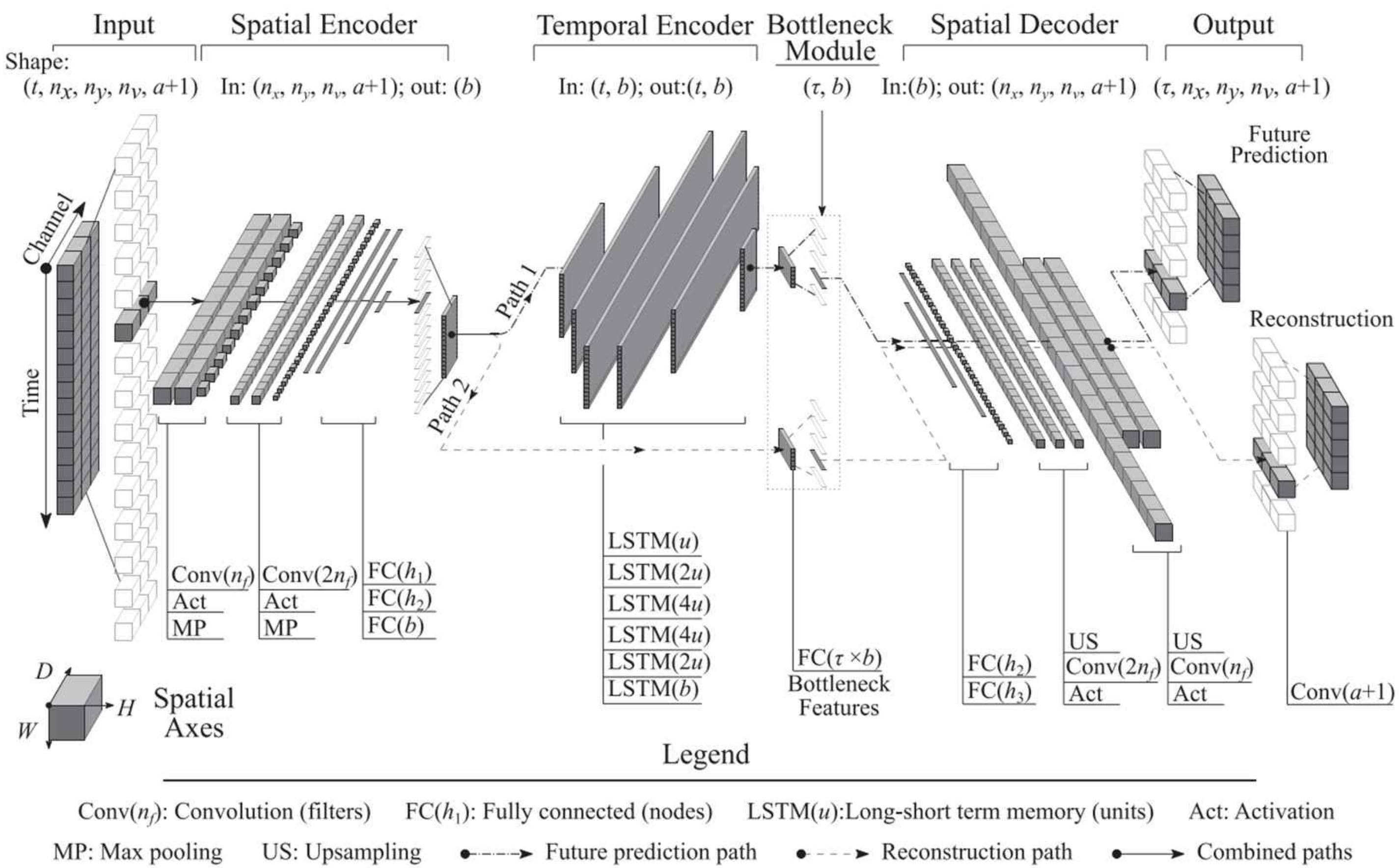
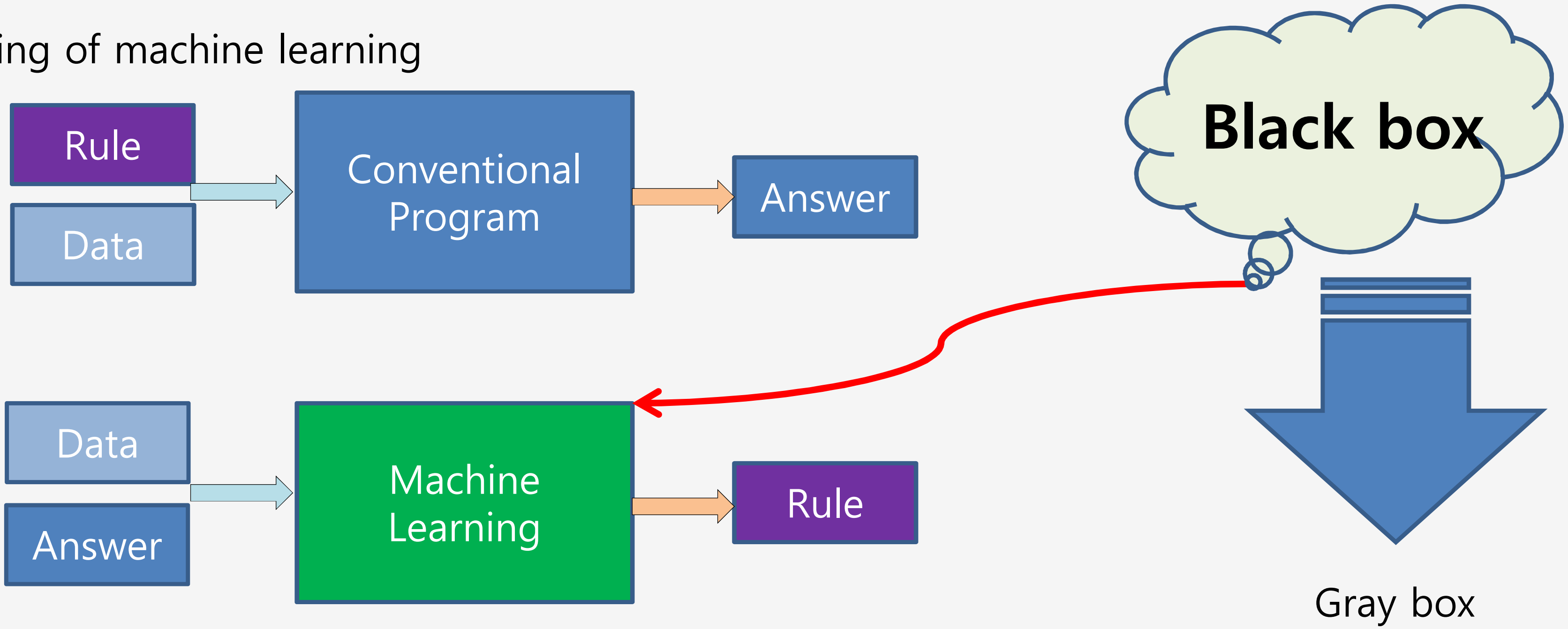


FIGURE 3 The proposed spatiotemporal composite autoencoder network architecture

Motivation

understanding of machine learning



Difference between traditional approach and ML

Contents

- Regression
- Matrix for Optimization
- Diagonalization
- Image Compression and Compressed sensing

- Data-driven Machine learning
- Physic-informed neural network
- Neural operator
- Kernelization

Basics in ML and DL

Deterministic model

$$y = f\left(x; \theta\right)$$

$$x \in X, y \in Y, \quad \text{and} \quad \theta = \left\{\theta_1, \cdots, \theta_D\right\}$$

Probabilistic model

Generative model distribution

$$p\left(x, y; \theta\right)$$

Discriminative model distribution

$$p\left(y \middle| x; \theta\right)$$

Summary of model functions and distribution

	Method	Model function/Distribution
Data fitting	Linear regression Nonlinear regression	$f(x;w) = w^T x$ $f(x;w) = \varphi(w,x)$
Artificial neural networks	Perception Feed-forward neural network Recurrent neural network Boltzmann machine	$f(x;w) = \varphi(w^T, x)$ $f(x;W_1,W_2,\cdots) = \cdots\varphi_2(W_2\varphi_1(W_1x))\cdots$ $f(x^{(t)};W) = \varphi(Wf(x^{(t-1)};W))$ $p(v;\theta) = \frac{1}{Z}\sum_h e^{-E(v,h;\theta)}$
Graphical models	Bayesian network Hidden Markov model	$p(s) = \prod_k p(s_k \vert \pi_k;\theta)$ $p(V,O) = \prod_{t=1}^{T} p(v^{(t)} \vert v^{(t-1)}) \prod_{t=1}^{T} p(o^{(t)} \vert v^{(t)})$
Kernel methods	Kernel density estimation K-nearest neighbor Support vector machine Gaussian Process	$p(x \vert y=c) = \frac{1}{M_c}\sum_{m \vert y^m = c}^{M} k(x - x^m)$ $p(x \vert y=c) = \frac{\# NN_c}{k}$ $f(x) = \sum_{m=1}^{M} \gamma_m y^m \kappa(x, x^m) + w_0$ $p(y \vert x) = N\left[\tilde{\kappa}^T K^{-1} y; \tilde{\tilde{\kappa}} - \tilde{\kappa}^T K^{-1} \tilde{\kappa}\right]$

Regression

$$\mathbf{Y} = f(\mathbf{X}, \beta)$$

The general relationship between independent variables $\mathbf{X}$, dependent variables $\mathbf{Y}$, and some unknown parameter β

Curve fitting results in an optimization problem.
The optimization can be mathematically framed as solving the linear system of equations

$$\mathbf{A}\,\mathbf{x} = \mathbf{b}$$

A simple solution for this linear problem uses the Moore-Penrose pseudo-inverse $\mathbf{A}^{\dagger}$.

$$\mathbf{x} = \mathbf{A}^{\dagger}\mathbf{b}$$

Linear Prediction

After training and validation

$$y_i = \sum_{j=1}^{d} x_{ij}\beta_j$$

$$\mathbf{y} = \mathbf{X}\beta$$

$$\mathbf{b} = \mathbf{A}\mathbf{x}$$

$\mathbf{y} \in \mathbb{R}^{n\times 1}$ **prediction**

$\mathbf{X} \in \mathbb{R}^{n\times d}$ Given data

$\boldsymbol{\beta} \in \mathbb{R}^{d\times 1}$ **parameter**

In matrix form

Given Data

$$\begin{bmatrix} t_1 \\ \bullet \\ \bullet \\ t_n \end{bmatrix} = \begin{bmatrix} x_{11} & \bullet & \bullet & x_{1d} \\ \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet \\ x_{n1} & \bullet & \bullet & x_{nd} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \bullet \\ \bullet \\ \beta_d \end{bmatrix}$$

Optimization ⇒

$$y_{predict} = \begin{Bmatrix} x_{new,1} \\ \bullet \\ \bullet \\ x_{new,d} \end{Bmatrix}^T \begin{bmatrix} \beta_1 \\ \bullet \\ \bullet \\ \beta_d \end{bmatrix}$$

prediction

label **Feature vector** **Learnable parameter**

MSE cost function for Linear regression model

$$\mathrm{MSE}(\mathbf{X}, h_\theta) = \frac{1}{m}\sum_{i=1}\left(\boldsymbol{\theta}^T \mathbf{x}^{(i)} - t^{(i)}\right)^2$$

Normal Equation to find the value of θ that minimizes the cost function

$$\hat{\boldsymbol{\theta}} = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{t}$$

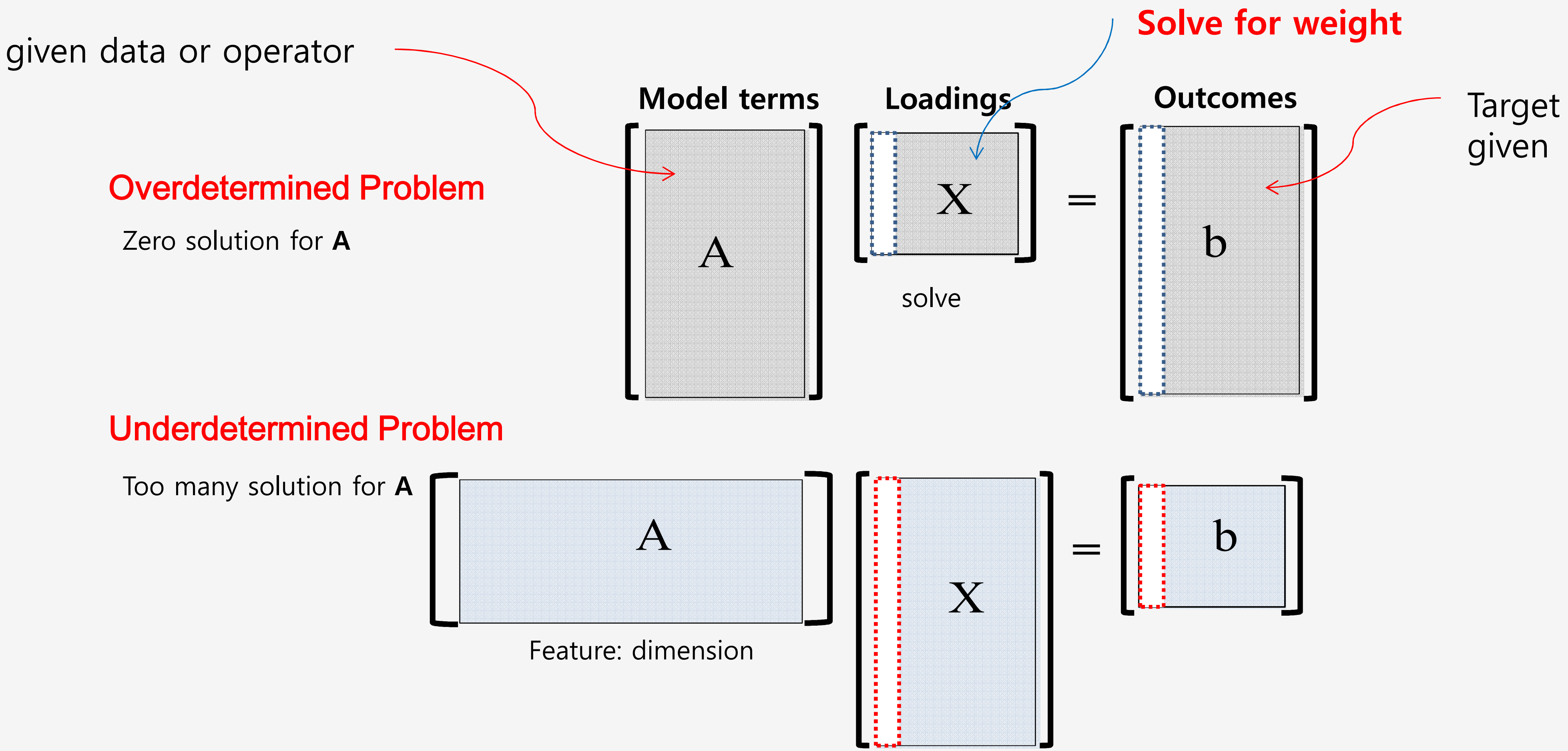
```
X_b = np.c_[np.ones((100, 1)), X]  # add x0 = 1 to each instance
theta_best = np.linalg.inv(X_b.T.dot(X_b)).dot(X_b.T).dot(y)
```

$$\hat{\boldsymbol{\theta}} = \mathbf{X}^{+}\mathbf{t}$$

$$\mathbf{X}^{+} = \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^T$$

Singular Value Decomposition

Matrix for Optimization



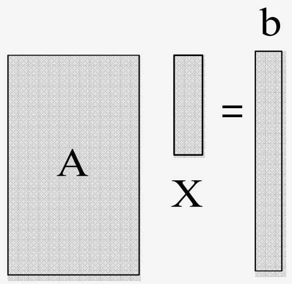
Solution X

$$\mathbf{A}\mathrm{x}=\mathrm{b}$$ Non-square A

How to deal with non-square matrices -> SVD, PCA,.....

Overdetermined, n>m (tall skinny A)

Many data



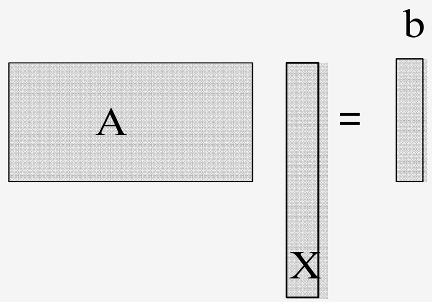
Zero solution X for given b

$$\min \|\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b}\|_2$$

Optimization

$$\operatorname{argmin}\left(\|\mathbf{Ax} - \mathbf{b}\|_2 + \lambda g(\mathbf{x})\right)$$

Underdetermined, n<m (short fat A)



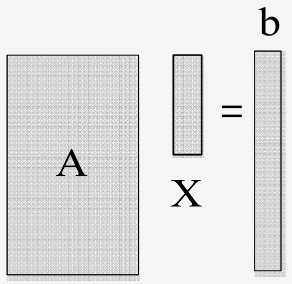
Too many solution x for given b

$$\min \|\tilde{\mathbf{x}}\|_2 \quad \text{such that } \mathbf{A}\tilde{\mathbf{x}}=\mathbf{b}$$

Optimization

$$\operatorname{argmin}\ g(\mathbf{x}) \quad \text{subject to } \|\mathbf{Ax} - \mathbf{b}\|_2 \le \varepsilon$$

Over-determined Systems



Zero solution X for given b

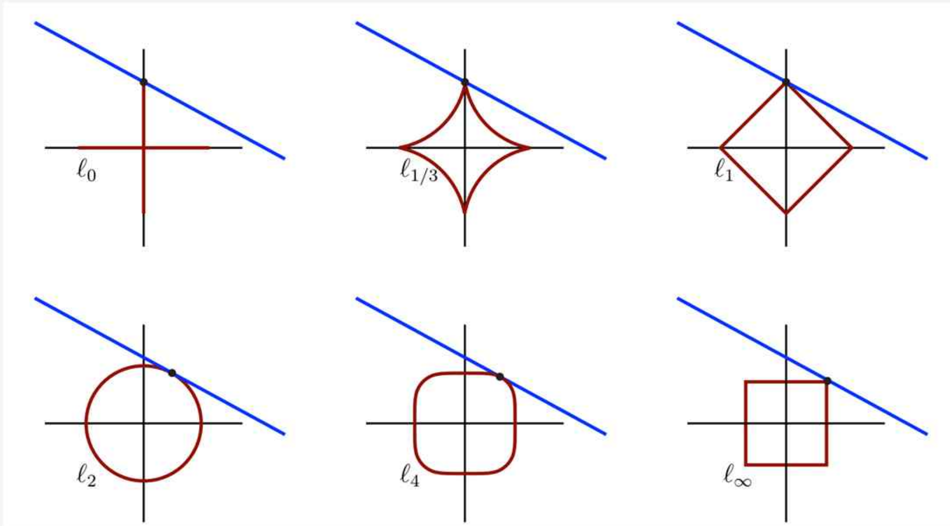
$$\min \|\mathbf{A}\tilde{\mathbf{x}} - \mathbf{b}\|_2$$

Optimization

$$\hat{\mathbf{x}} = \operatorname{argmin} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 + \lambda_1 \|\mathbf{x}\|_1 + \lambda_2 \|\mathbf{x}\|_2$$

l_2 norm   Lasso   Ridge

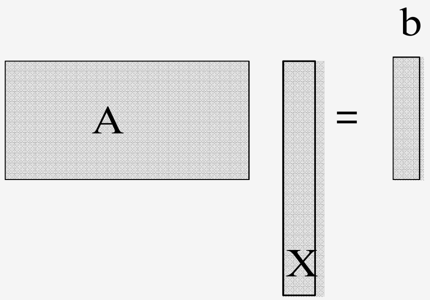
Least absolute shrinkage and selection operator

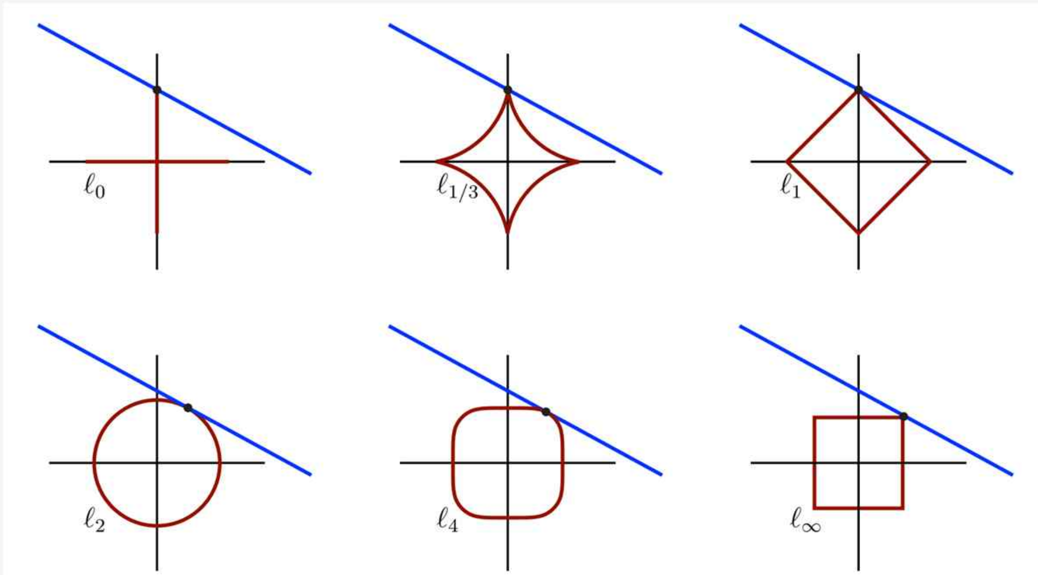


The minimum-norm point on a line in different lp norm

From Fig 3.10, p 110 in Ref.1

Under-determined Systems





Too many solution x for given b

$$\min \|\tilde{\mathbf{x}}\|_p \quad \text{subject to } \mathbf{A}\tilde{\mathbf{x}}=\mathbf{b}$$

Optimization

$$\min \left(\lambda_1 \|\mathbf{x}\|_1 + \lambda_2 \|\mathbf{x}\|_2\right) \quad \text{subject to } \|\mathbf{A}\mathbf{x}-\mathbf{b}\|_2 \le \varepsilon$$

Diagonalization (Square matrix)

$$\underline{\underline{A}}\,\underline{v} = \lambda \underline{v}$$

Eigen vector

Eigen value

$$\mathbf{A} = \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{-1}$$

$$\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \mathbf{\Lambda}$$

$$[\mathbf{A}] = \begin{bmatrix} 5 & 1 \\ 3 & 3 \end{bmatrix} \Rightarrow |\mathbf{A} - \lambda\mathbf{I}| = 0$$

$$\vec{\mathbf{v}}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \lambda_1 = 6 \qquad \vec{\mathbf{v}}_2 = \begin{bmatrix} 1 \\ -3 \end{bmatrix}, \lambda_2 = 2$$

$$[\mathbf{A}] = \begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix} \Rightarrow |\mathbf{A} - \lambda\mathbf{I}| = 0$$

$$\vec{\mathbf{v}}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \lambda_1 = 4 \qquad \vec{\mathbf{v}}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \lambda_2 = 2$$

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \Rightarrow \mathbf{A}^{-1} = \frac{1}{ad - cb}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

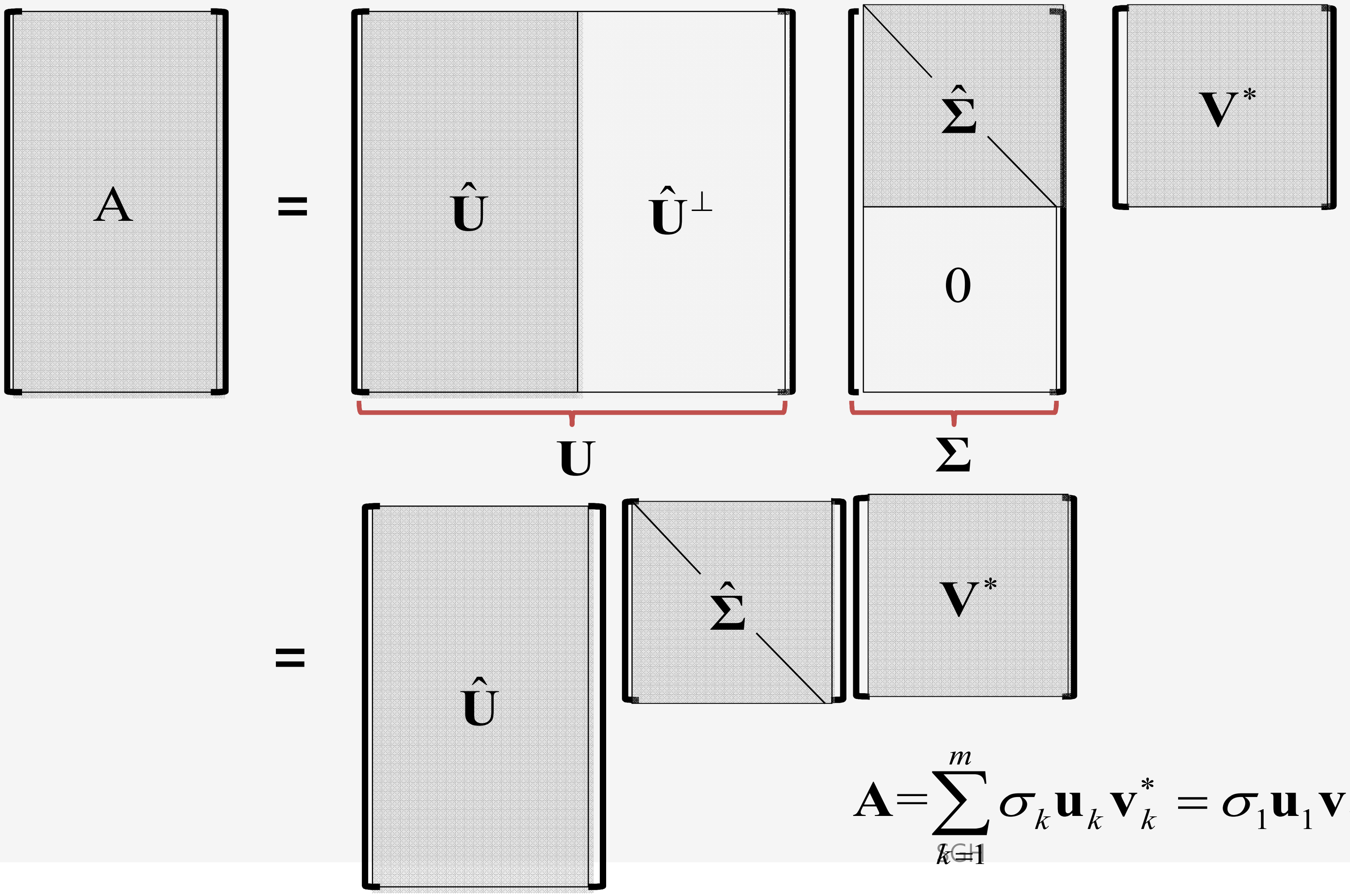
$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ 1 & -3 \end{bmatrix} \qquad \mathbf{X}^{-1} = \frac{1}{4}\begin{bmatrix} 3 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \frac{1}{4}\begin{bmatrix} 3 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 5 & 1 \\ 3 & 3 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ 1 & -3 \end{bmatrix} = \begin{bmatrix} 6 & 0 \\ 0 & 2 \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \qquad \mathbf{X}^{-1} = \frac{1}{2}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} = \mathbf{X}^T$$

$$\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \frac{1}{2}\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 3 & -1 \\ -1 & 3 \end{bmatrix}\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix}$$

Rectangular matrix decomposition(SVD)



$$\mathbf{A} = \sum_{k=1}^{m} \sigma_k \mathbf{u}_k \mathbf{v}_k^* = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^* + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^* + \cdots + \sigma_m \mathbf{u}_m \mathbf{v}_m^*$$

Image Compression and Compressed Sensing

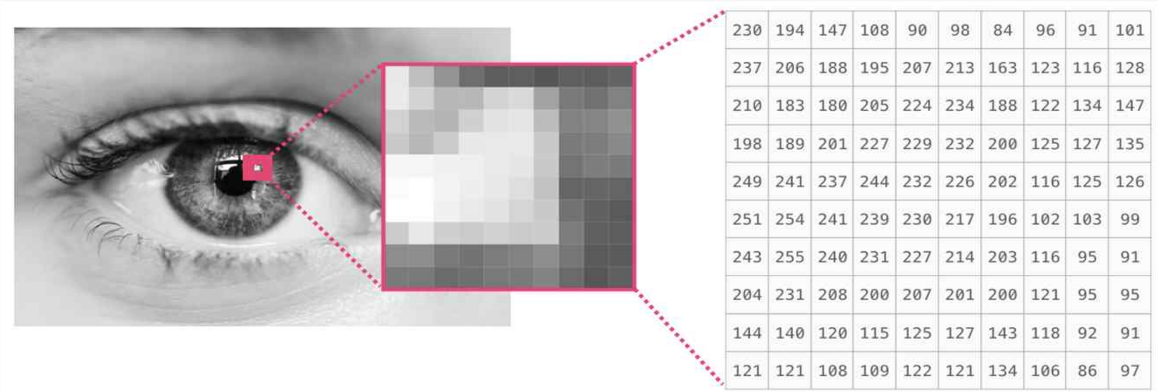
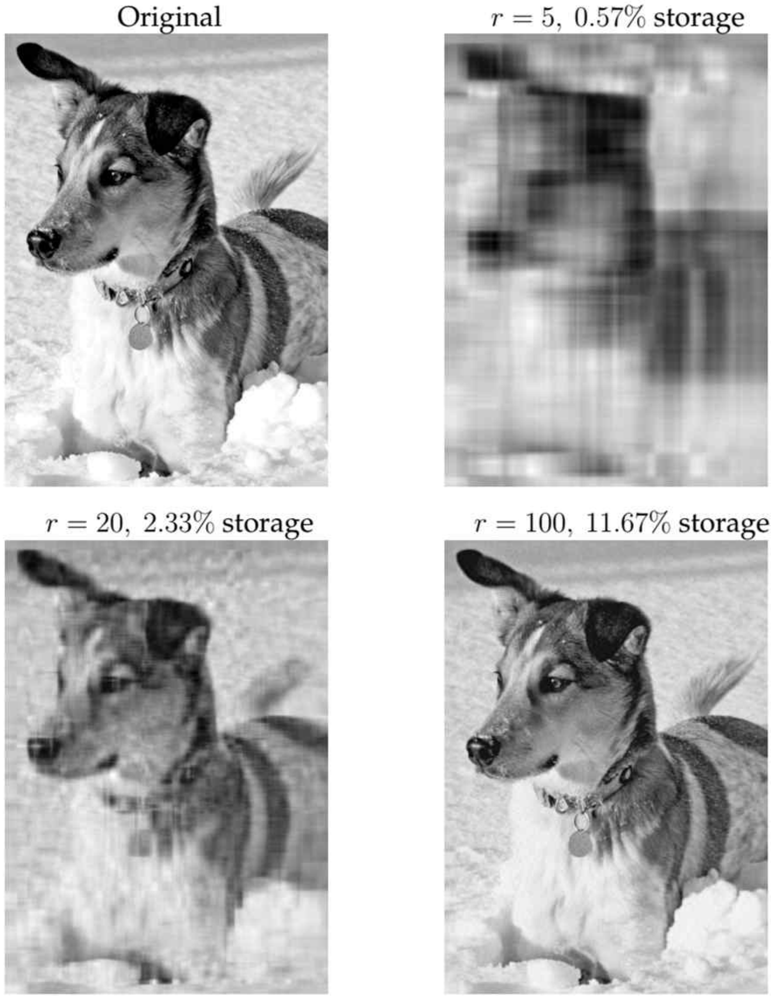


Image Compression



$$\mathbf{X} = \sum_{k=1}^{m} \sigma_k \mathbf{u}_k \mathbf{v}_k^* = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^* + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^* + \cdots + \sigma_m \mathbf{u}_m \mathbf{v}_m^*$$

$$\mathbf{X} - \tilde{\mathbf{X}} = \sum_{k=r+1}^{m} \sigma_k \mathbf{u}_k \mathbf{v}_k^*$$

Image compression truncating the SVD at various rank r
Original image resolution is 2000 X 1500

From Fig 1.3, p 110 in Ref.1

2D Fourier Transformation for Image

$$\hat{f}(\omega) = F(f(x)) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$$

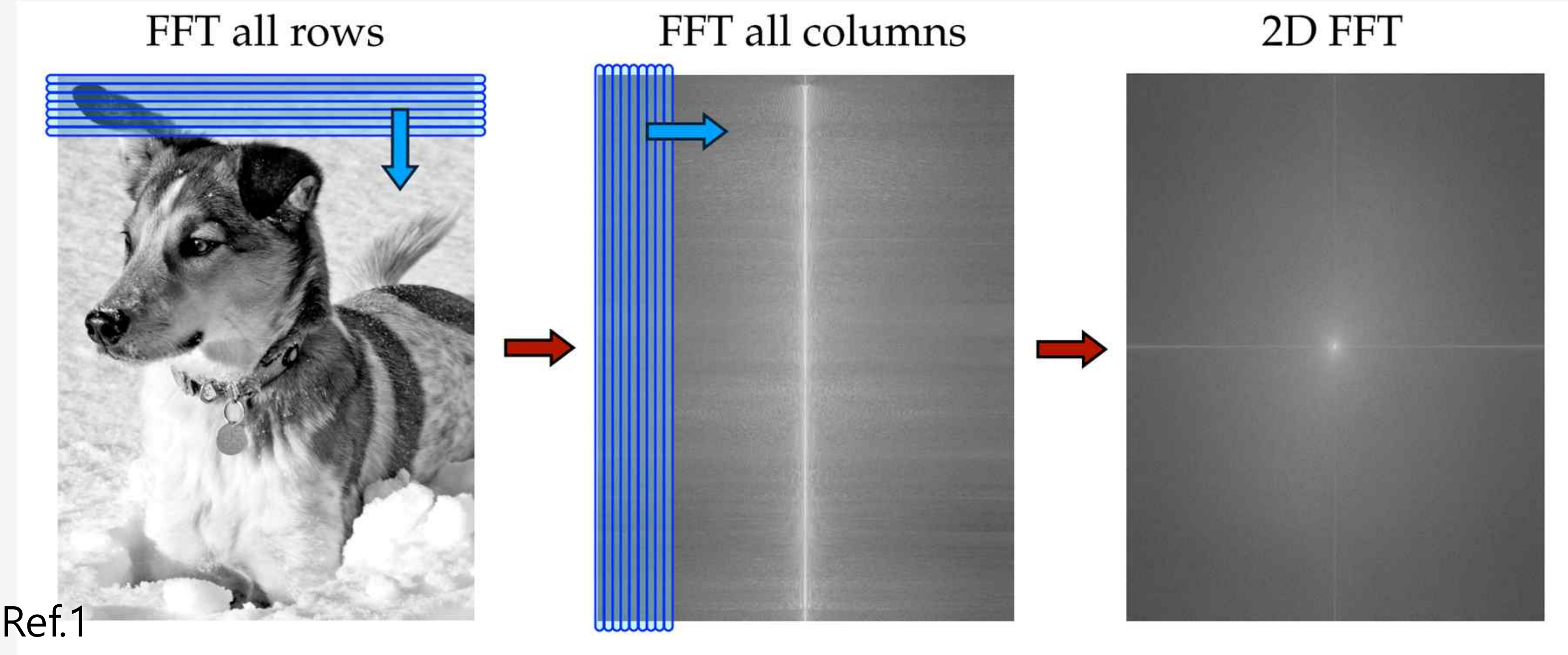
$$f(x) = F^{-1}(\hat{f}(\omega)) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega x} d\omega$$

$$\hat{f}(u,v) = F(f(x,y)) = \int_{-\infty}^{\infty} f(x,y) e^{-i\omega(ux+vy)} dxdy$$

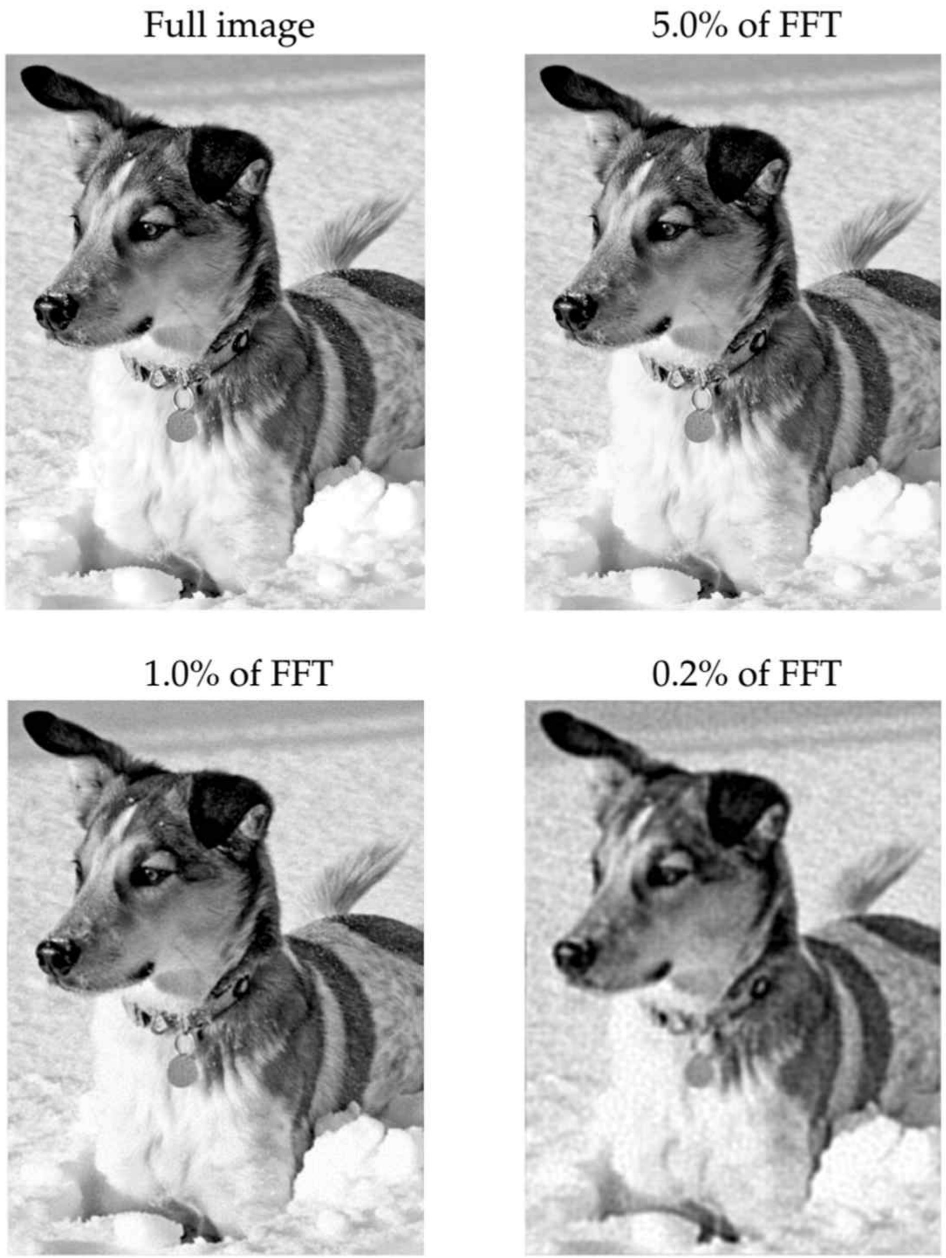
$$f(x,y) = F^{-1}(\hat{f}(u,v)) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(u,v) e^{i\omega(ux+vy)} dudv$$

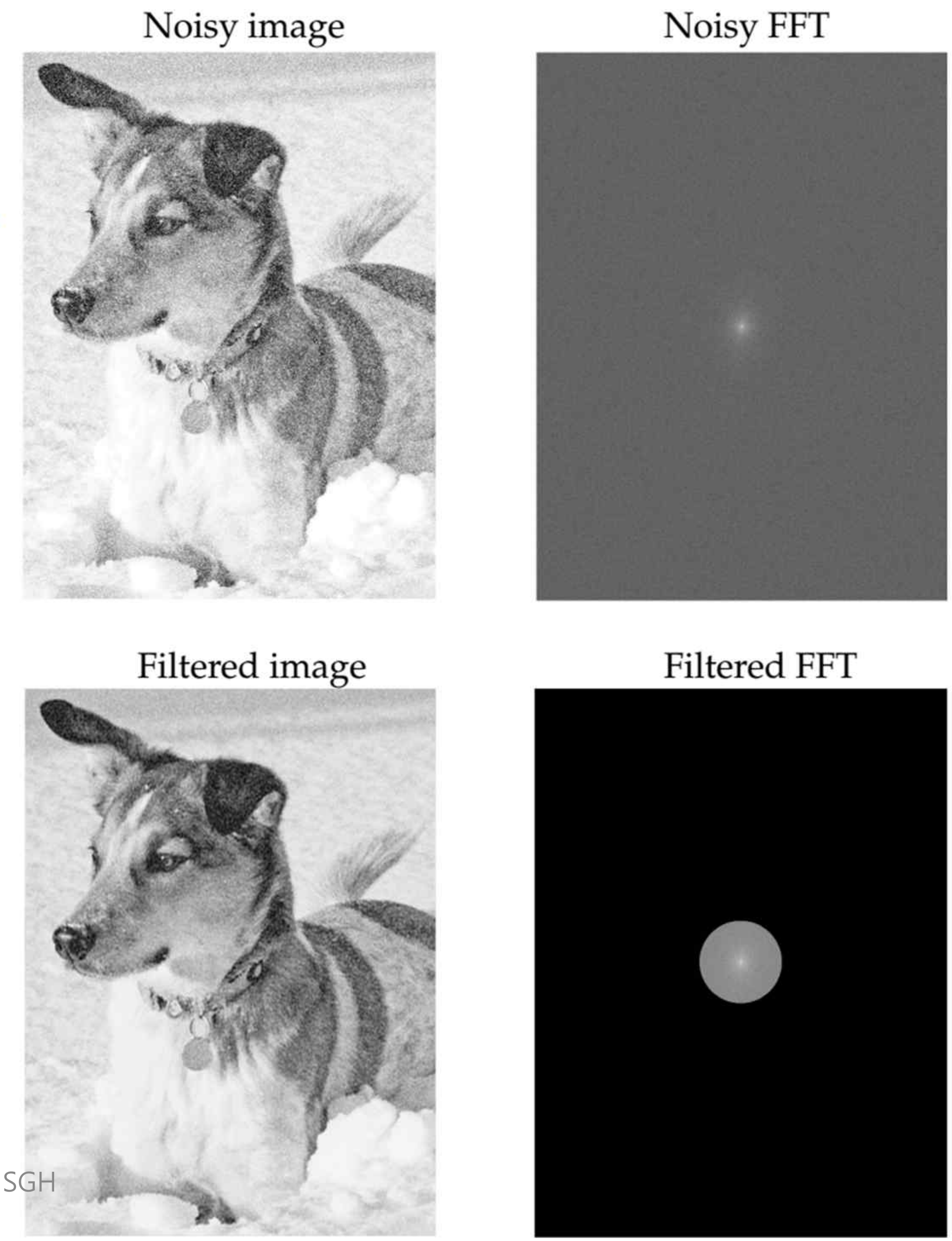
Image compression truncating Fourier Transformation

Original image resolution is 2000 X 1500

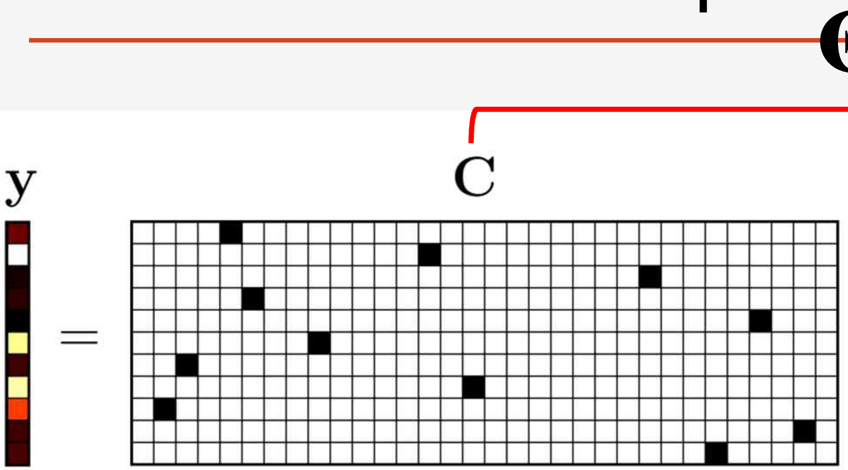


From Fig 1.3, p 110 in Ref.1


Full image
5.0% of FFT
1.0% of FFT
0.2% of FFT

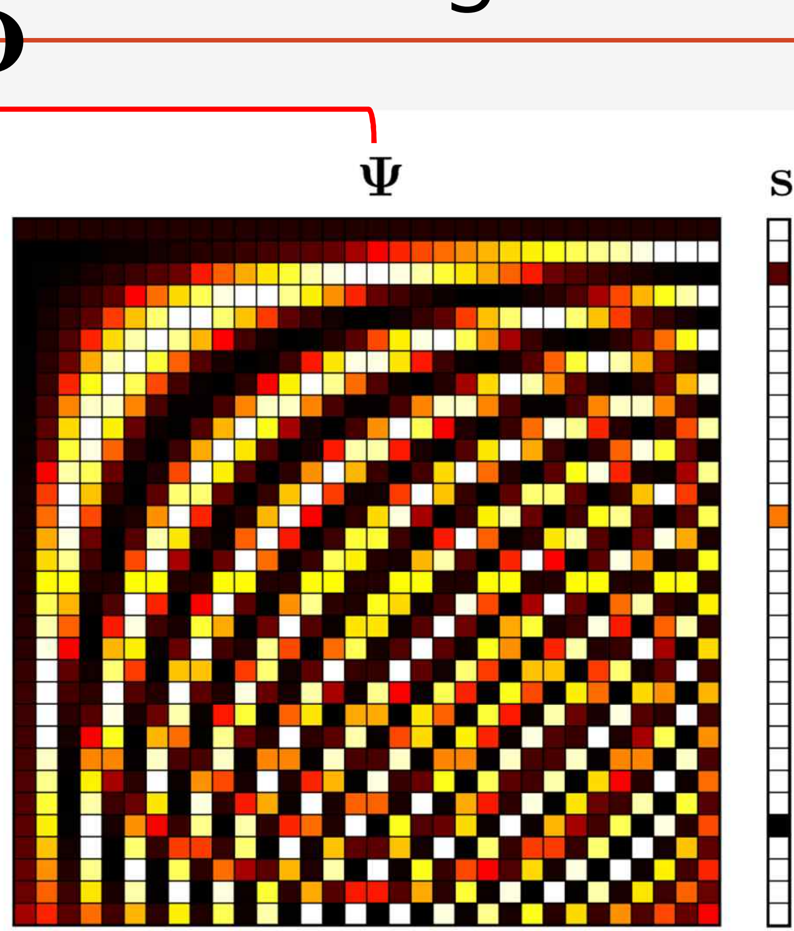

Noisy image
Noisy FFT
Filtered image
Filtered FFT

Compressed sensing



measurement

Measurement matrix



Orthonormal basis

Sparse vector

Schematic of measurement in compressed sensing framework

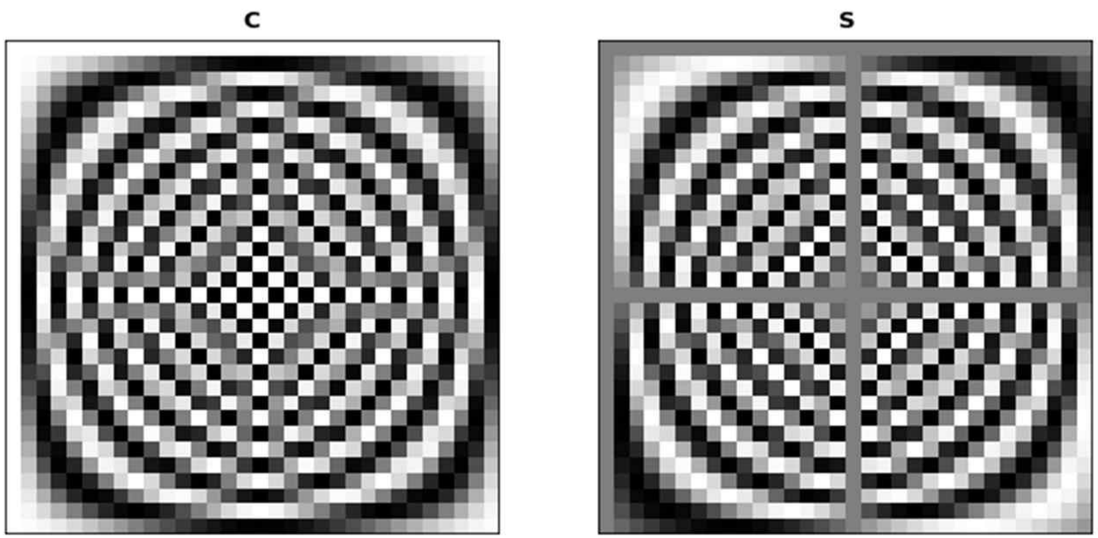
From Fig 3.4, p 103 in Ref.1

$$\mathbf{y} = \mathbf{C}\mathbf{x}$$

$$\mathbf{y} = \mathbf{C}\mathbf{\Psi}\mathbf{s} = \mathbf{\Theta}\mathbf{s}$$

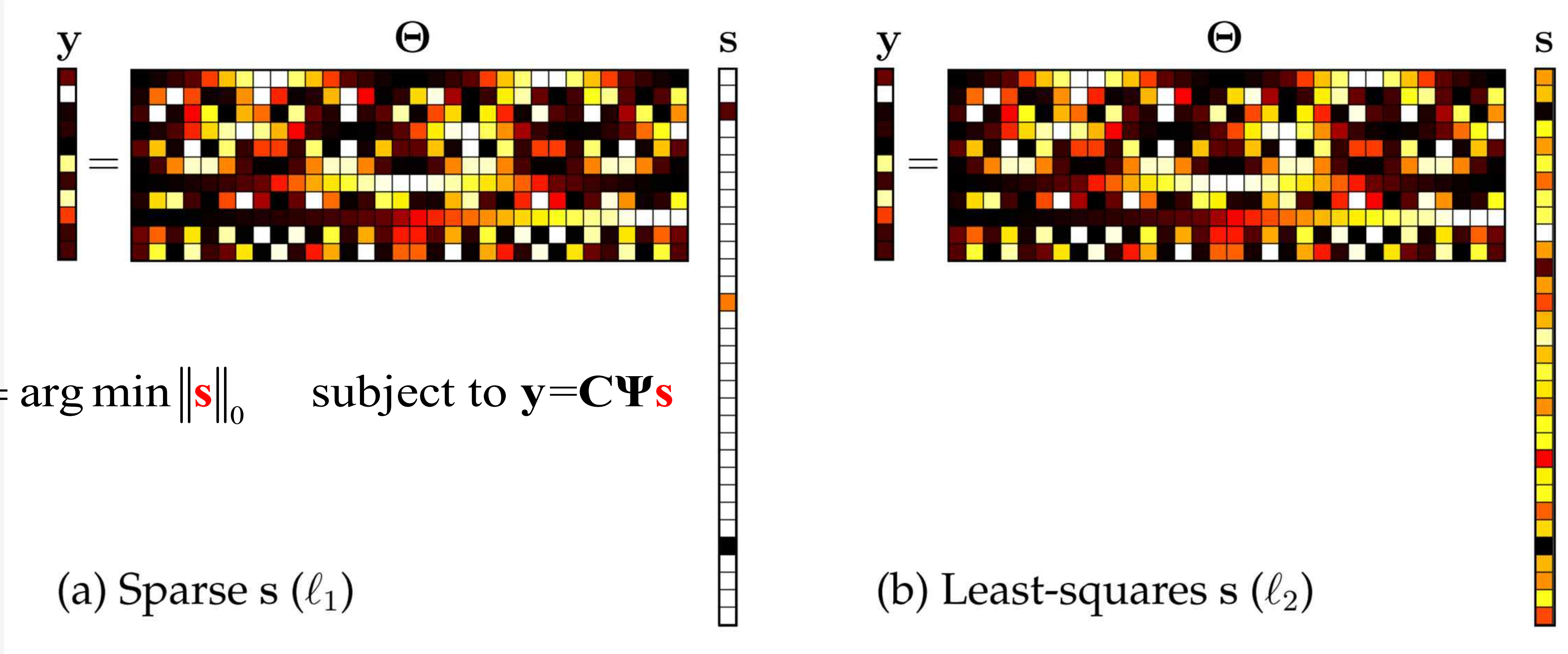
$$\hat{\mathbf{s}} = \arg\min \|\mathbf{s}\|_0 \quad \text{subject to } \mathbf{y}=\mathbf{C}\mathbf{\Psi}\mathbf{s}$$

Fourier Basis; Orthonormal basis



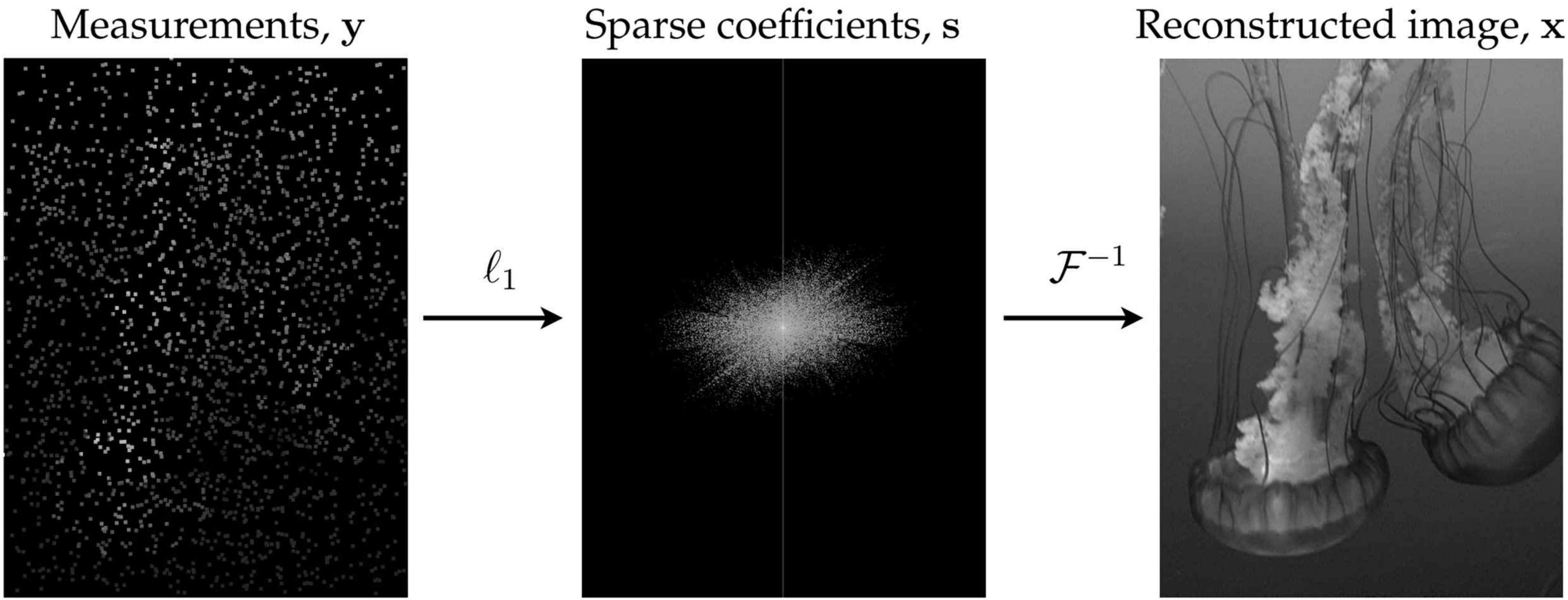
[The Fourier basis — Tutorials on imaging, computing and mathematics (matthew-brett.github.io)]

Solutions dependent on norm selection



$$\hat{\mathbf{s}} = \arg\min \|\mathbf{s}\|_0 \quad \text{subject to } \mathbf{y} = \mathbf{C\Psi s}$$

From Fig 3.5, p 103 in Ref.1



Measurement

Sparse coefficient

Reconstructed image

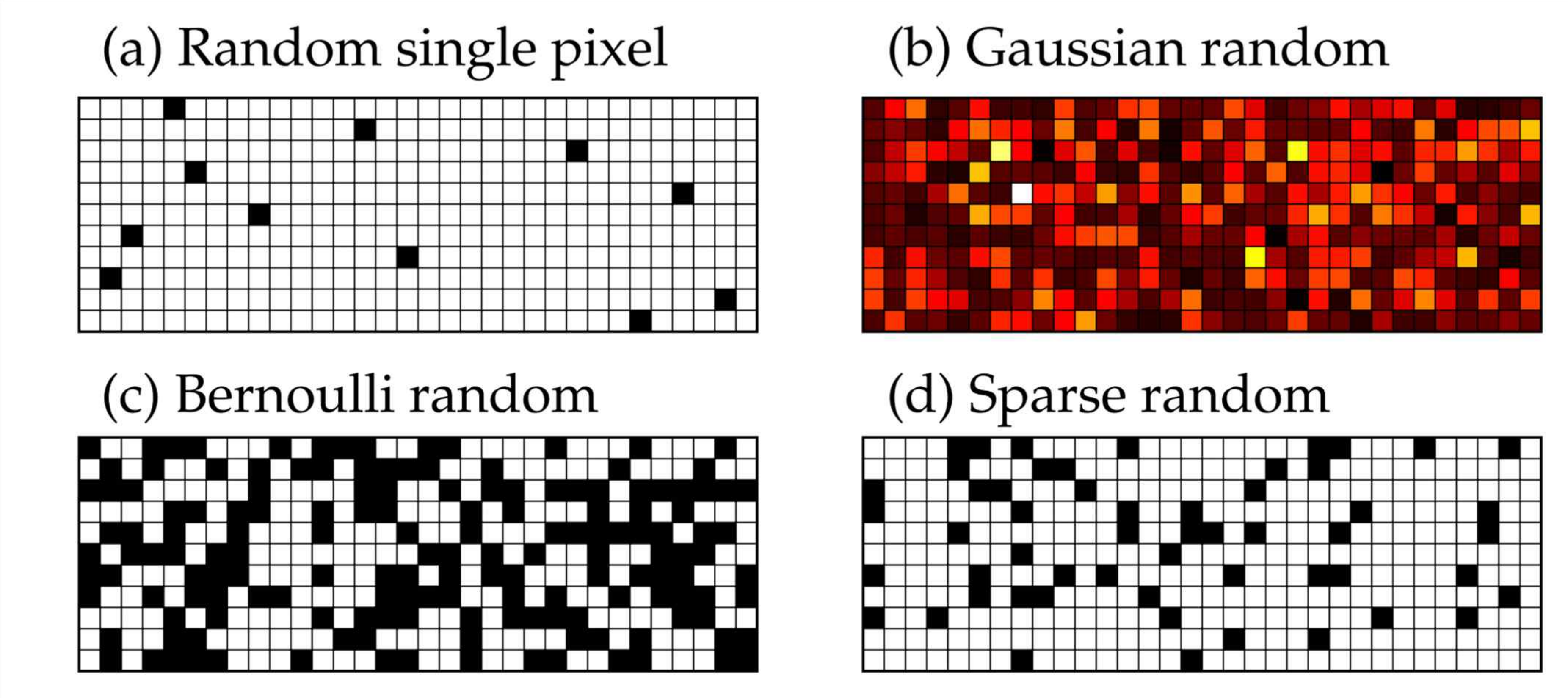
$$p \approx O\left(K \ln\left(n / K\right)\right) \approx k_1 K \ln\left(n / K\right)$$

$$p \approx 3 * 0.05 * \underbrace{1024 * 768}_{\text{Pixel : n}} * \ln\left(20\right) = 353,390$$

$$K = 0.05 * 1024 * 768$$

Pixel : n

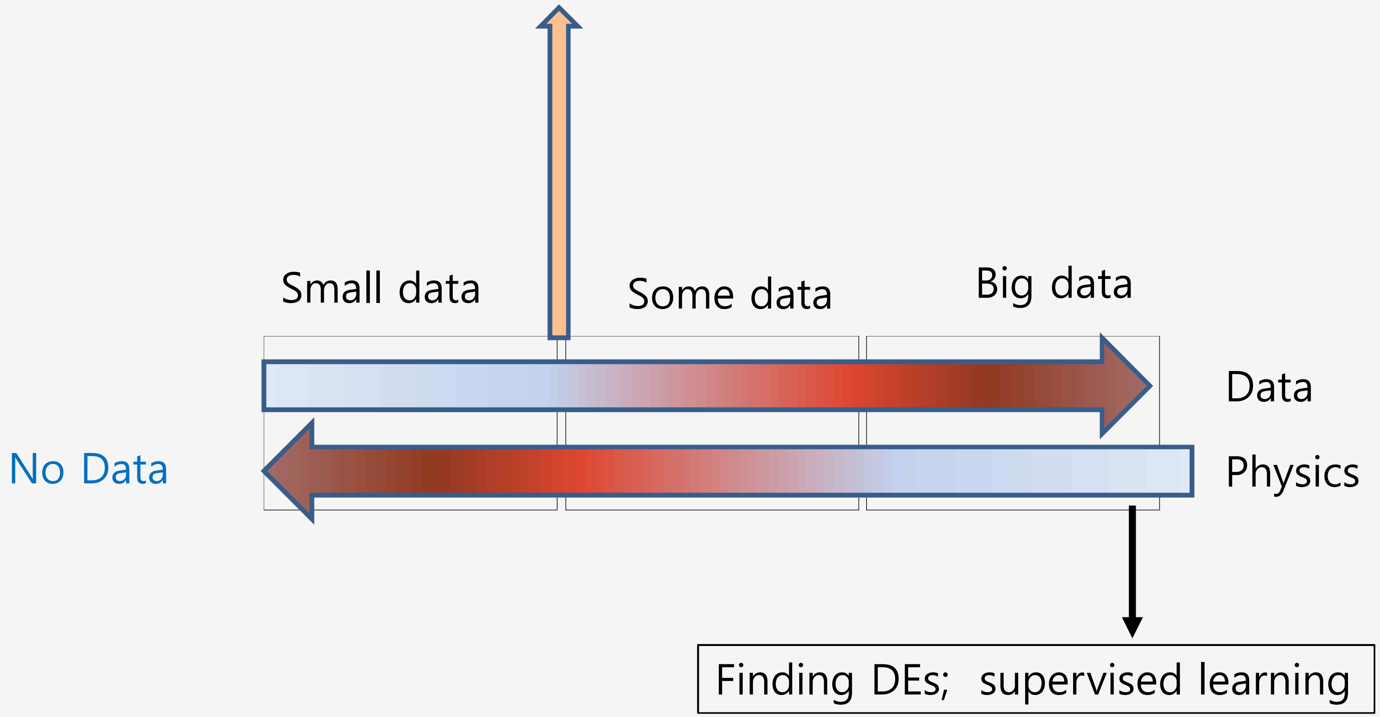
Measurement matrices



Examples of good random measurement matrices

From Fig 3.11, p 112 in Ref.1

Data-Driven Machine Learning



Agenda for Engineering with AI

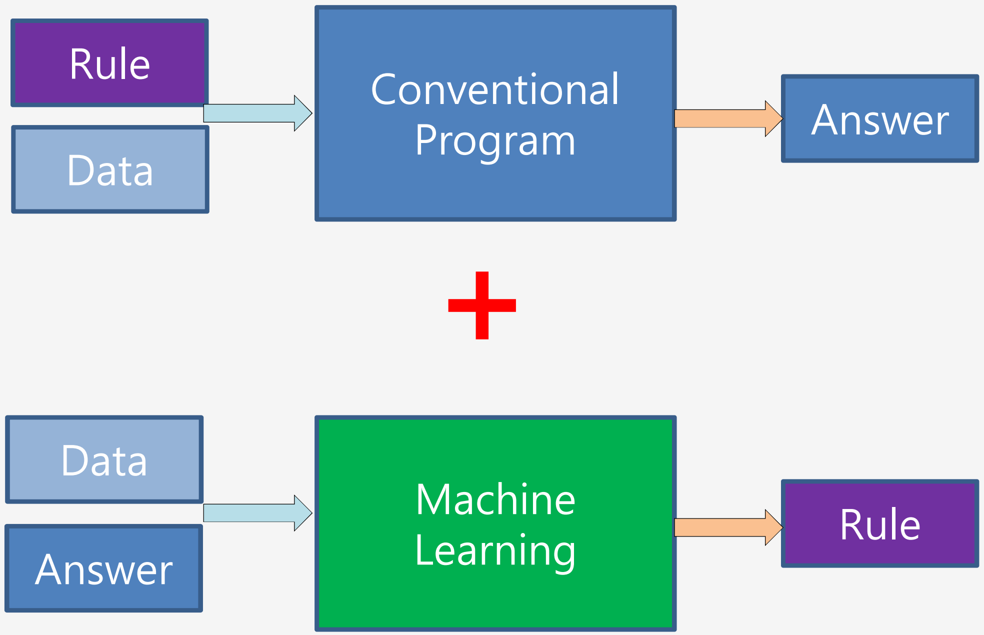
PHYSICS INFORMED MACHINE LEARNING

- Derivation of Governing Equations
  Data-driven M/L
- Solution of Governing Equations (PDE, DE)
  Physics-informed M/L

PHYSICAL MODELS FROM DATA via OPTIMIZATION

- How to optimize by a few data
- Data-driven M/L -> to find linear operator

Paradigm shift

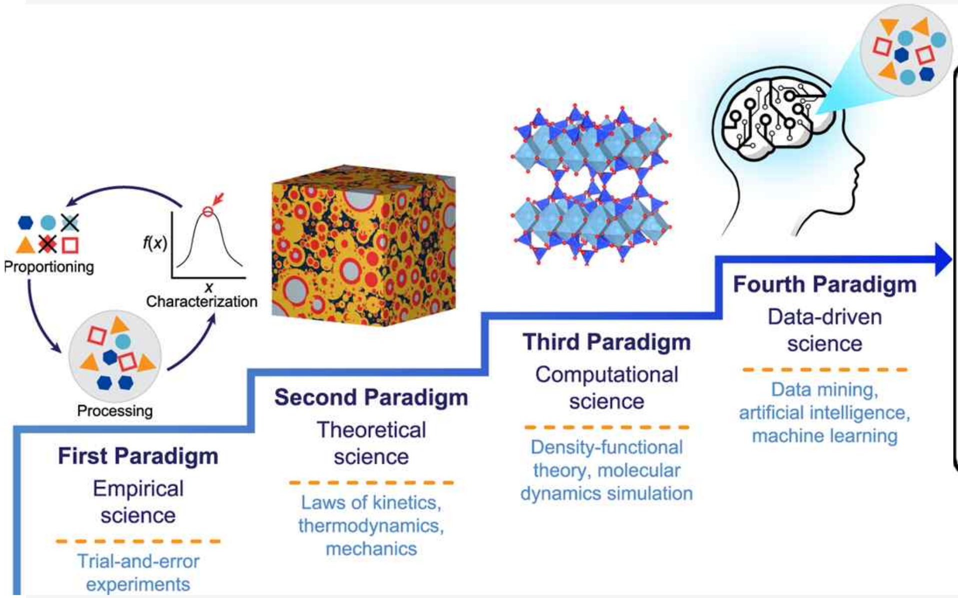


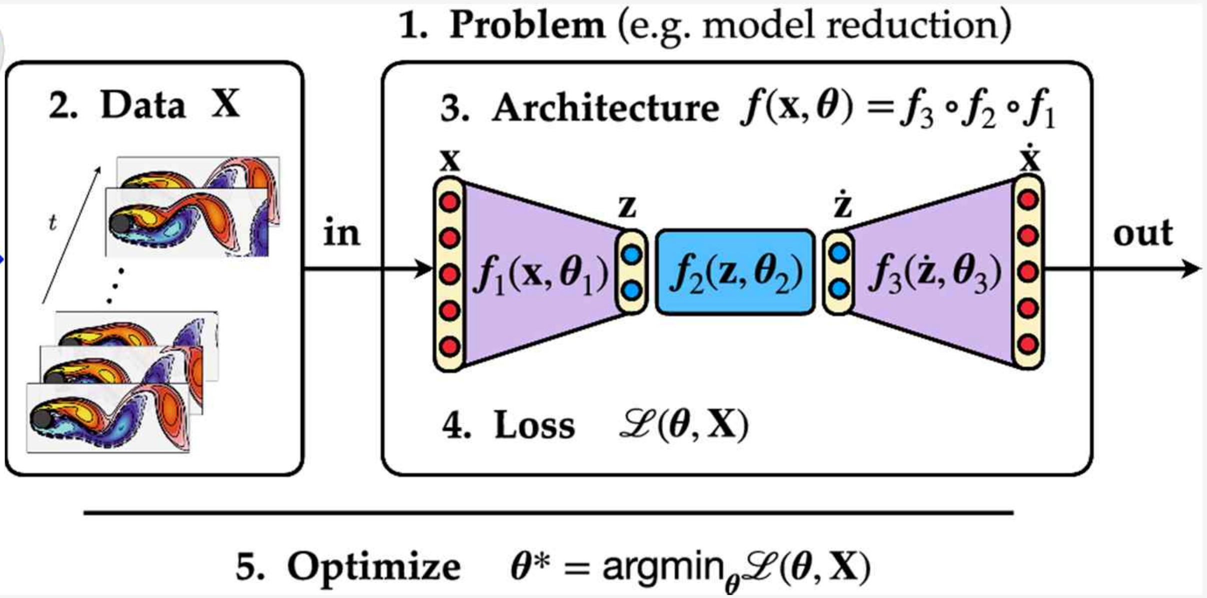
Combination between traditional approach and ML

Application area of Data-driven ML

ML for new material

Fluid mechanics





https://media.springernature.com/full/springer-static/image/art%3A10.1038%2Fs41524-022-00810-x/MediaObjects/41524_2022_810_Fig1_HTML.png?as=webp

Applying machine learning to study fluid mechanics
Invited Review
Open access
Published: 04 January 2022
Volume 37, pages 1718–1726, (2021

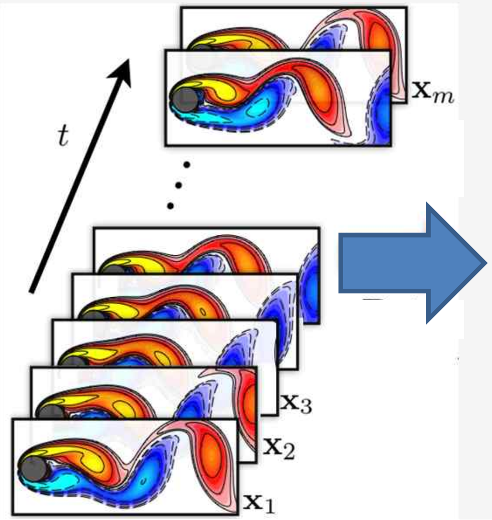
Dynamic system

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}\left(\mathbf{x}(t), t; \beta\right)$$

Vector field

State of system

Set of parameters



Linear dynamics and Spectral Decomposition

$$\frac{d}{dt}\mathbf{x} = \mathbf{A}\mathbf{x} \quad \Rightarrow \quad \mathbf{x}(t_0 + t) = e^{\mathbf{A}t}\mathbf{x}(t_0)$$

The dynamics are entirely characterized by the eigenvalues of A, given by the spectral decomposition

$$\mathbf{A}\mathbf{T} = \mathbf{T}\mathbf{\Lambda}$$

$$\mathbf{A} = \mathbf{T}\mathbf{\Lambda}\mathbf{T}^{-1}$$

$$\mathbf{x}(t_0 + t) = \mathbf{T}e^{\mathbf{\Lambda}t}\mathbf{T}^{-1}\mathbf{x}(t_0)$$

Transformation of coordinate gives decoupled system

$$\mathbf{z} = \mathbf{T}^{-1}\mathbf{x}$$

$$\frac{d}{dt}\mathbf{z} = \mathbf{\Lambda}\mathbf{z}$$

Dynamic Mode Decomposition

$$\mathbf{X} = \begin{bmatrix} | & | & & & | \\ \mathbf{x}(t_1) & \mathbf{x}(t_2) & \cdot\cdot & \cdot\cdot & \mathbf{x}(t_m) \\ | & | & & & | \end{bmatrix}$$

$$\mathbf{X}' = \begin{bmatrix} | & | & & & | \\ \mathbf{x}\left(t_1'\right) & \mathbf{x}\left(t_2'\right) & \cdot\cdot & \cdot\cdot & \mathbf{x}\left(t_m'\right) \\ | & | & & & | \end{bmatrix}$$

$$\mathbf{X}' \approx \mathbf{AX}$$

pseudo-inverse .

The best-fit operator $$\mathbf{A} = \underset{A}{\arg\min} \left\| \mathbf{X}' - \mathbf{AX} \right\|_F = \mathbf{X}'\mathbf{X}^\dagger$$

Steps for DMD

Step 1. Compute the singular value decomposition of X

$$\mathbf{X} \approx \tilde{\mathbf{U}}\tilde{\boldsymbol{\Sigma}}\tilde{\mathbf{V}}^*$$

reduced singular vector, conjugate transpose

Step 2. The reduced-order matrix A

$$\mathbf{A} = \mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\mathbf{U}}^* \Rightarrow \tilde{\mathbf{A}} = \tilde{\mathbf{U}}^*\mathbf{A}\tilde{\mathbf{U}} = \tilde{\mathbf{U}}^*\mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}$$

Step 3. The spectral decomposition of the reduced matrix A

$$\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\boldsymbol{\Lambda}$$

Step 4. The high-dimensional DMD modes are reconstructed

$$\boldsymbol{\Phi} = \mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\mathbf{W}$$

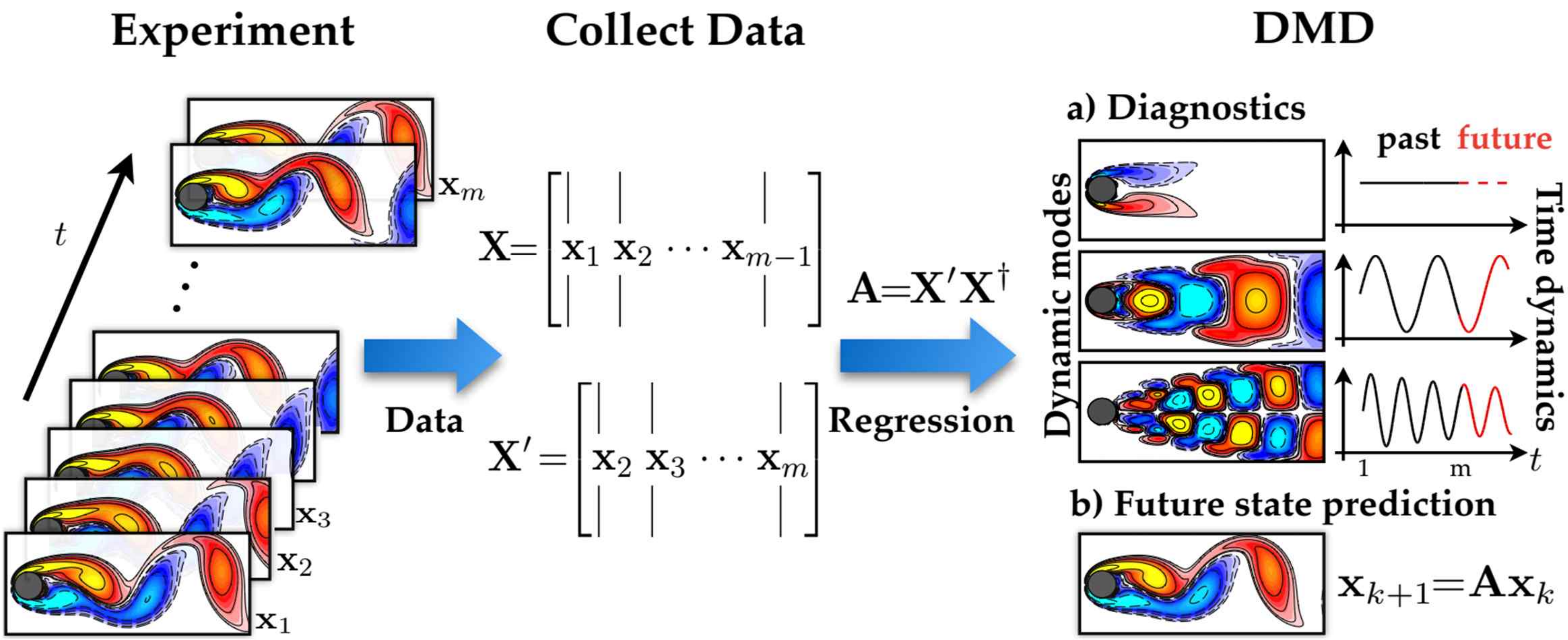
$$\begin{aligned}
\mathbf{A}\boldsymbol{\Phi} &= \underbrace{\left(\mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\mathbf{U}}^*\right)}_{\mathbf{A}}\underbrace{\left(\mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\mathbf{W}\right)}_{\boldsymbol{\Phi}} \quad (\tilde{\mathbf{U}}^*\mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1} = \tilde{\mathbf{A}}) \\
&= \mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\mathbf{A}}\mathbf{W} \\
&= \mathbf{X}'\tilde{\mathbf{V}}\tilde{\boldsymbol{\Sigma}}^{-1}\mathbf{W}\boldsymbol{\Lambda} \\
&= \boldsymbol{\Phi}\boldsymbol{\Lambda}
\end{aligned}$$

Algorithm 1 Exact DMD [4]

Input: Data matrix $\boldsymbol{X}$, shifted data matrix $\boldsymbol{X}'$, and target rank r.
Output: DMD spectrum $\boldsymbol{\Lambda}$ and modes $\boldsymbol{\Phi}$.

1: **procedure** DMD $(\boldsymbol{X}, \boldsymbol{X}', r)$
2: $[\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}] \leftarrow \mathrm{SVD}(\boldsymbol{X}, r)$ ▷ Truncated r-rank SVD of $\boldsymbol{X}$.
3: $\tilde{\boldsymbol{A}} \leftarrow \boldsymbol{U}^* \boldsymbol{X}' \boldsymbol{V} \boldsymbol{\Sigma}^{-1}$ ▷ Low-rank approximation of $\boldsymbol{A}$.
4: $[\boldsymbol{W}, \boldsymbol{\Lambda}] \leftarrow \mathrm{EIG}(\tilde{\boldsymbol{A}})$ ▷ Eigendecomposition of $\tilde{\boldsymbol{A}}$.
5: $\boldsymbol{\Phi} \leftarrow \boldsymbol{X}' \boldsymbol{V} \boldsymbol{\Sigma}^{-1} \boldsymbol{W}$ ▷ DMD modes of $\boldsymbol{A}$.
6: **end procedure**

Note that if $\lambda_i = 0$, then $\boldsymbol{\phi}_i = \boldsymbol{U}\boldsymbol{w}_i$ for step 5. In the original DMD algorithm [57] all modes are computed as $\boldsymbol{\phi}_i = \boldsymbol{U}\boldsymbol{w}_i$.



Overview of DMD illustrated on the fluid flow past a circular cylinder at Reynolds number 100. From Ref. 1

Sparse Identification of Non-linear Dynamics

$$\frac{d}{dt}\mathbf{x}(t) = \mathbf{f}\left(\mathbf{x}(t), t; \beta\right)$$

$$\mathbf{X} = \left[\mathbf{x}(t_1) \quad \mathbf{x}(t_2) \quad \cdots \quad \cdots \quad \mathbf{x}(t_m)\right]^T$$

We seek to approximate **f** by a generalized linear model

$$\dot{\mathbf{X}} = \left[\dot{\mathbf{x}}(t_1) \quad \dot{\mathbf{x}}(t_2) \quad \cdots \quad \cdots \quad \dot{\mathbf{x}}(t_m)\right]^T$$

$$\mathbf{f}(\mathbf{x}) \approx \sum_{k=1}^{p} \theta_k(\mathbf{x})\boldsymbol{\xi} = \boldsymbol{\Theta}(\mathbf{x})\boldsymbol{\xi}$$

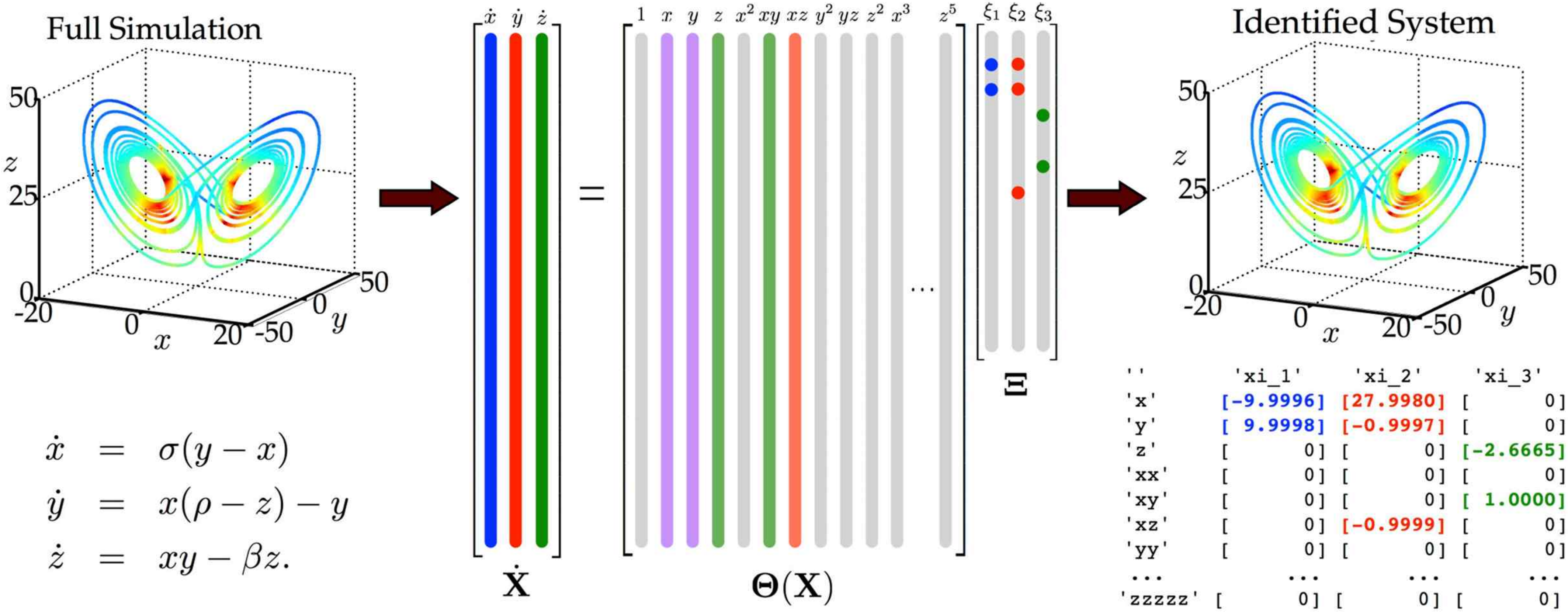
A library of **candidate nonlinear function** may be constructed from the data in X

$$\boldsymbol{\Theta}(\mathbf{X}) = \left[\mathbf{1} \quad \mathbf{X} \quad \mathbf{X}^2 \quad \cdots \quad \mathbf{X}^d \quad \cdots \quad \sin(\mathbf{X}) \quad \cdots\right]$$

$$\dot{\mathbf{X}} = \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}$$

A parsimonious model will provide an accurate model fit with as few terms as possible in $\boldsymbol{\Xi}$

$$\boldsymbol{\xi}_k = \arg\min_{\xi'_k} \left\|\dot{\mathbf{X}}_k - \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\xi}'_k\right\|_2 + \lambda\left\|\boldsymbol{\xi}'_k\right\|_1$$



$$\dot{\mathbf{X}} = \boldsymbol{\Theta}(\mathbf{X})\boldsymbol{\Xi}$$

$$\boldsymbol{\Theta}(\mathbf{X}) = \begin{bmatrix} \mathbf{1} & \mathbf{X} & \mathbf{X}^2 & \cdots & \mathbf{X}^d & \cdots & \sin(\mathbf{X}) & \cdots \end{bmatrix}$$

Discovering Partial Differential Equations

A major extension of SINDy modeling framework generalized the library to include partial derivative, enabling the identification of partial differential equation.

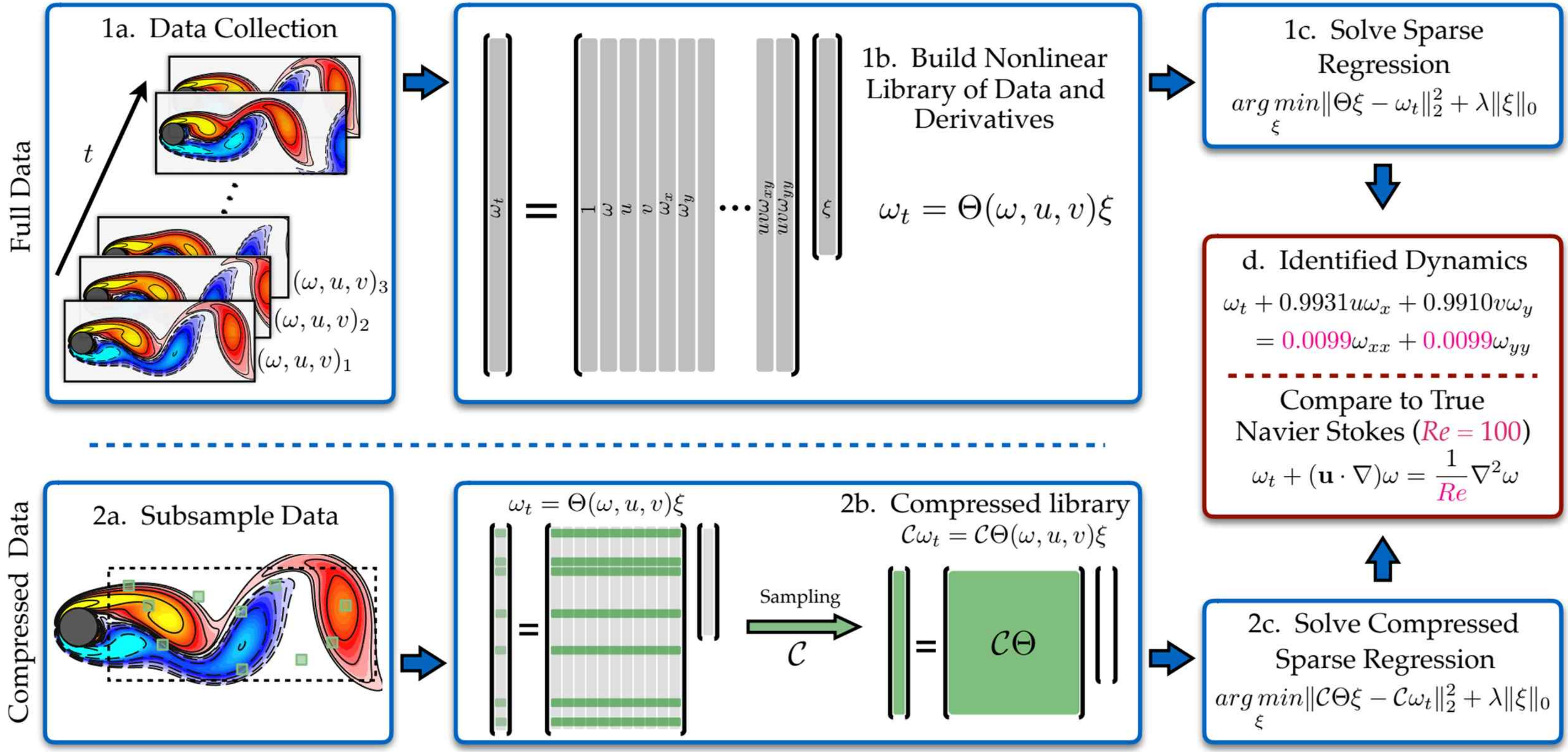
$$\boldsymbol{\Theta}(\Upsilon, \mathbf{Q}) = \begin{bmatrix} \mathbf{1} & \Upsilon & \Upsilon^2 & \cdots & Q & \cdots & \Upsilon_x & \Upsilon\Upsilon_x \cdots \end{bmatrix}$$

Spatial time-series

A known potential or magnitude of complex data

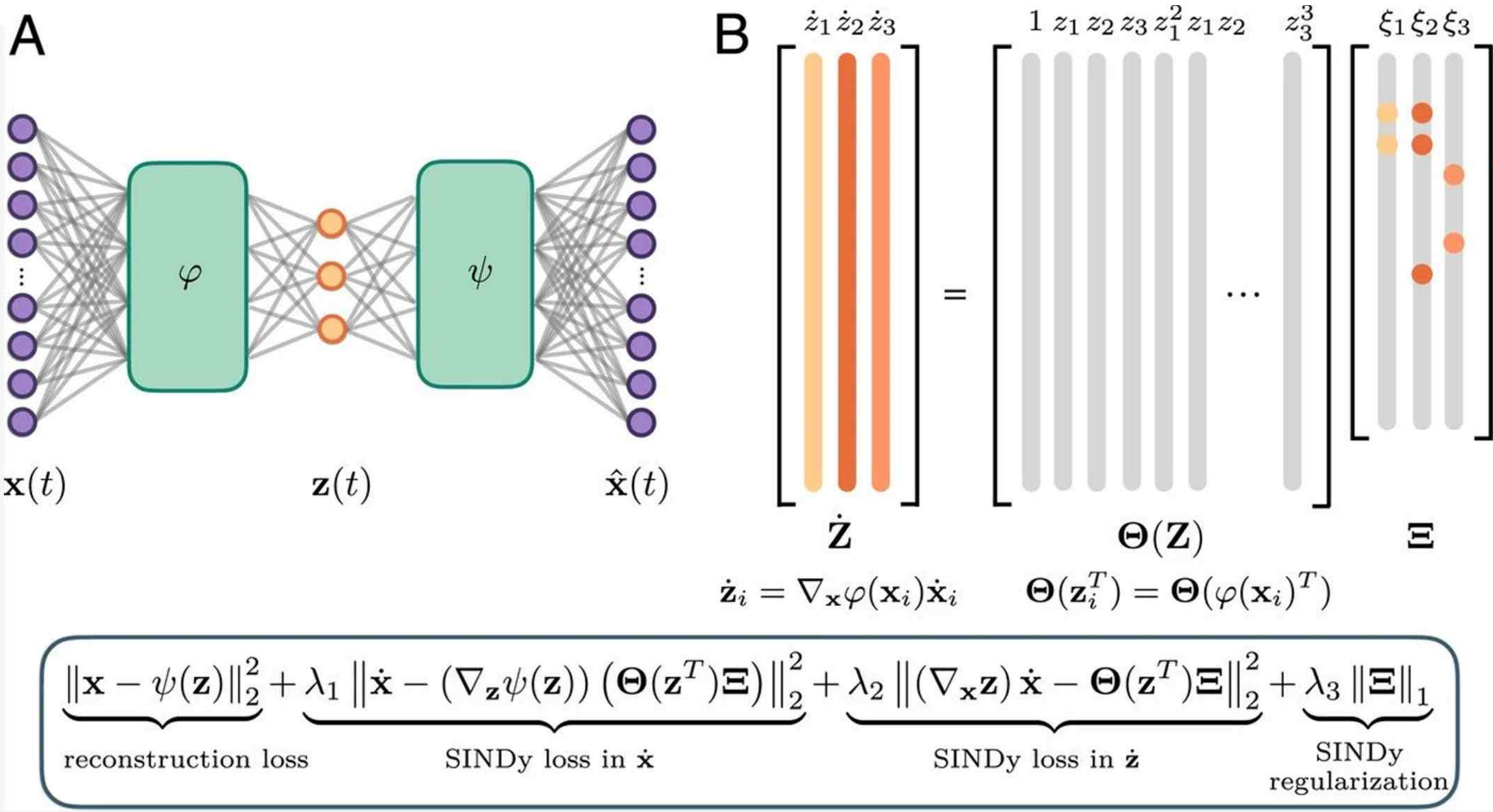
$$\Upsilon_t = \boldsymbol{\Theta}(\Upsilon, \mathbf{Q})\boldsymbol{\xi}$$

$$\hat{\boldsymbol{\xi}} = \arg\min_{\xi} \left\| \boldsymbol{\Theta}(\Upsilon, \mathbf{Q})\boldsymbol{\xi}'_k - \Upsilon_t \right\|_2^2 + \varepsilon\kappa\left(\Theta(\Upsilon, \mathbf{Q})\right)\left\|\boldsymbol{\xi}'\right\|_0$$



Data-driven discovery of coordinates and governing equations

Kathleen Champion1 , Bethany Lusch2, J. Nathan Kutz1, Steven L. Brunton



Schematic of the SINDy autoencoder method for simultaneous discovery of coordinates and parsimonious dynamics

https://www.pnas.org/cms/10.1073/pnas.1906995116/asset/71b09e6c-9d0f-4a9f-b1e5-9cc8d8375808/assets/graphic/pnas.1906995116fig01.jpeg

Application to structural vibration

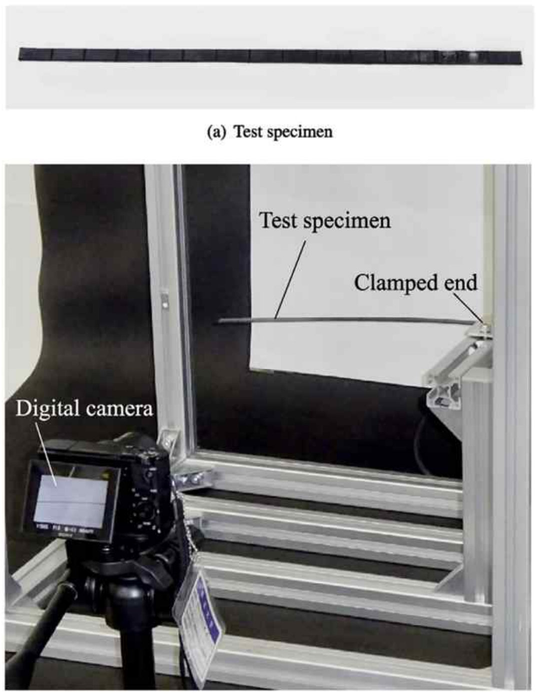
The application of Dynamic Mode Decomposition(DMD) to the extraction of modal properties of linear mechanical systems, i.e., experimental modal analysis (EMA).

Data-driven experimental modal analysis by Dynamic Mode Decomposition

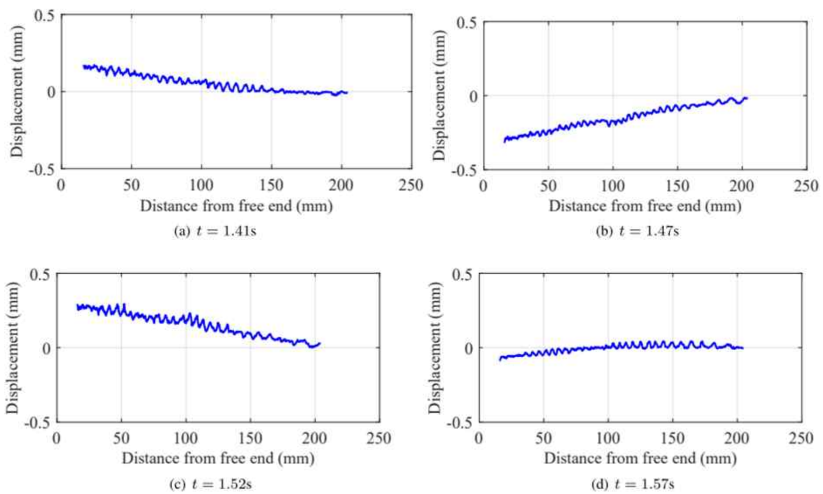
Akira Saito*, Tomohiro Kuno

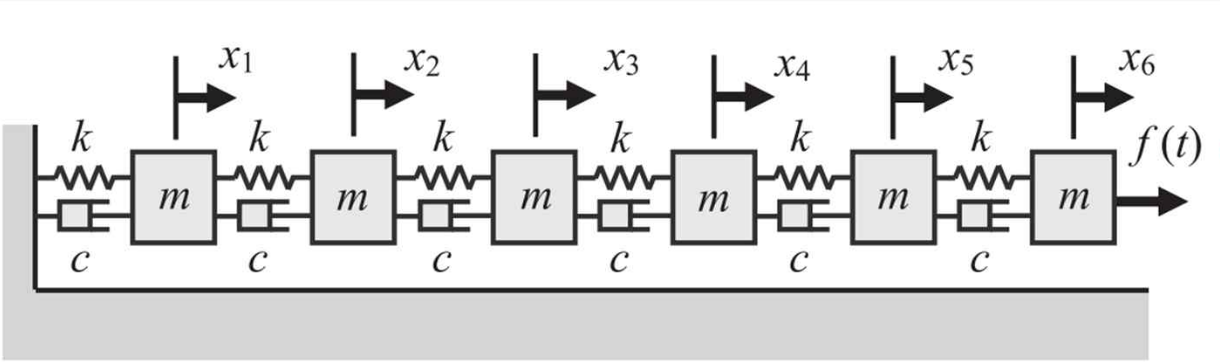
https://doi.org/10.1016/j.jsv.2020.115434 0022-460X/

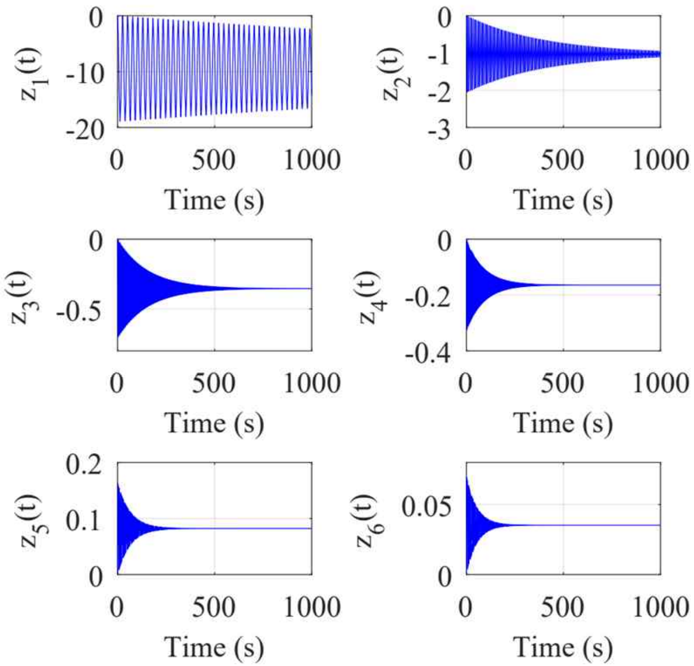
Journal of Sound and Vibration



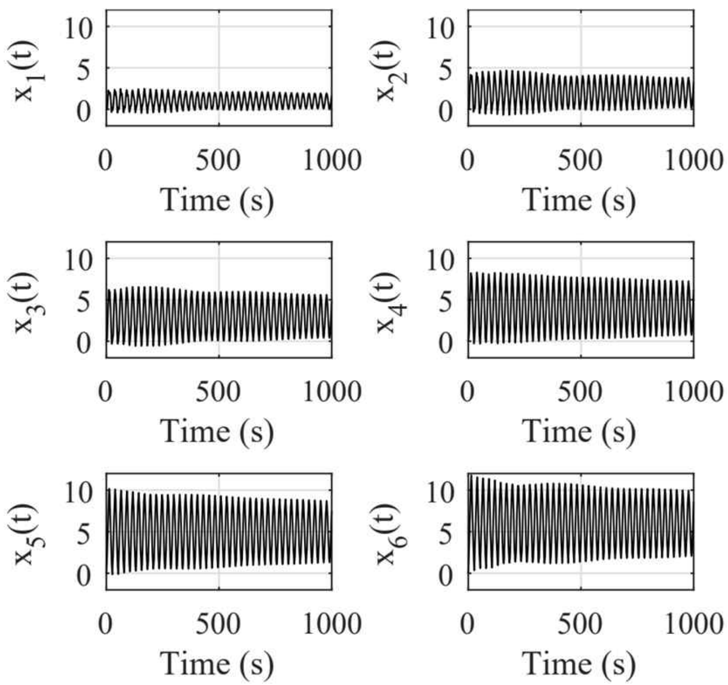
(b) Experimental setup







(a) Time response of modal coordinates ($\mathbf{z}(t)$)



(b) Time response of physical DOF ($\mathbf{x}(t)$)

Application of SINdy to Structural Dynamics

Sparse structural system identification method for nonlinear dynamic systems with hysteresis/inelastic behavior

By Zhilu Lai, Satish Nagarajaiah

Mechanical Systems and Signal Processing 117 (2019) 813–842

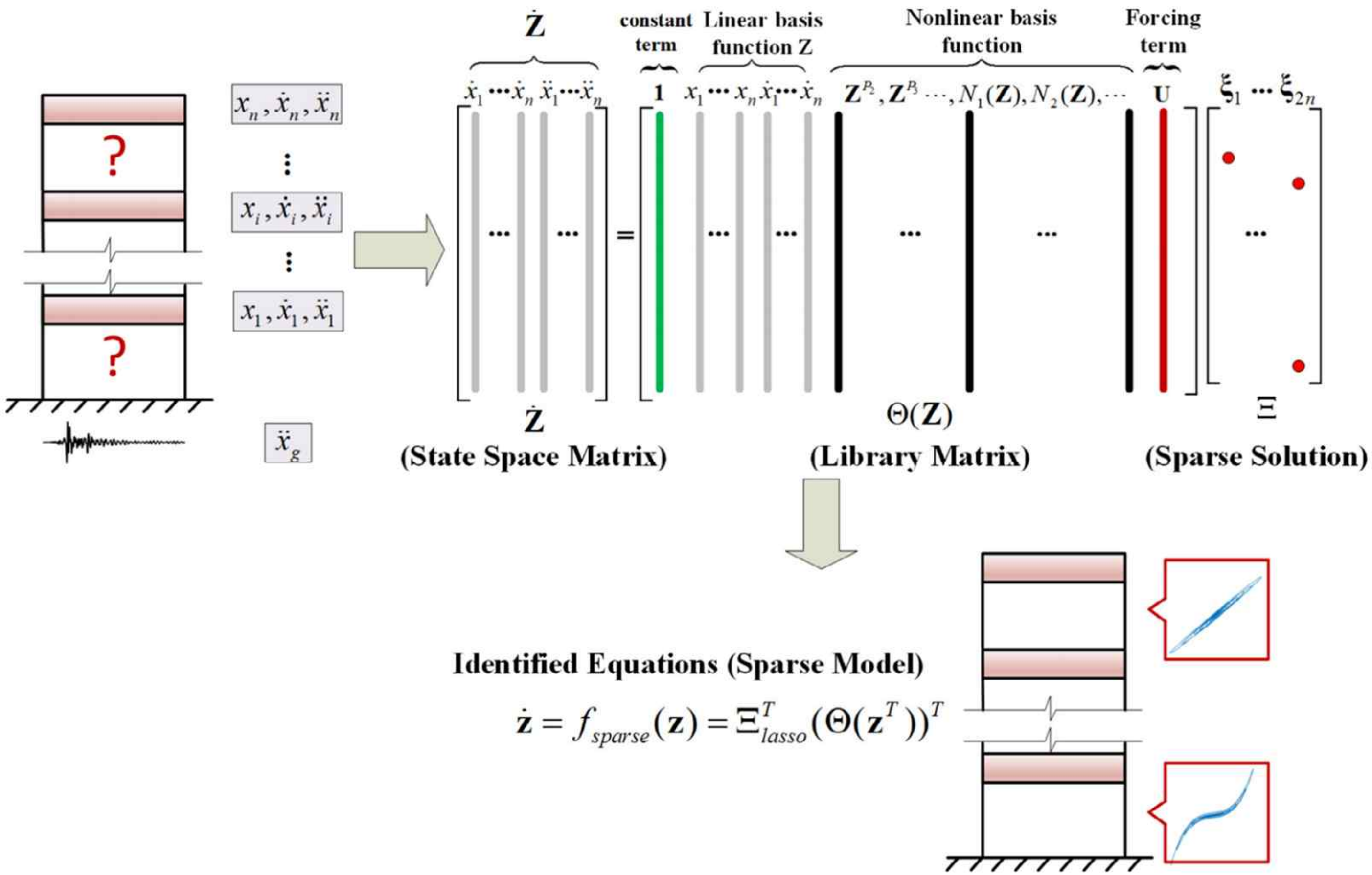


Fig. 2. Schematic procedure of sparse identification to nonlinear elastic structural systems.

Steps in SINDy for Dynamics

1. **Data Preparation**
   Obtain the measured data $x_i(t), \ddot{x}_i(t)$ and $\ddot{x}_g(t)$ at time $t_1, t_2, \ldots, t_m$ and compute $\dot{x}_i(t)$, and $\dot{a}_{h,i}(t)$
2. **Assemble $\mathbf{Z}_a$ and $\mathbf{Z}_h$**
   Depending on the choice of types of nonlinearities (nonlinear elastic or nonlinear inelastic), $\mathbf{Z}_a$ and $\mathbf{Z}_h$ are constructed through Eq. (30) and Eq. (32) for nonlinear inelastic behavior (or Eq. (4) for nonlinear elastic).
3. **Construct Library Matrix**
   Library matrix is constructed with only measured data, according to the choice of types of basis functions used to represent different nonlinearities as in Eq. (35) for nonlinear inelastic or Eq. (6) for nonlinear elastic.
4. **Sparse Feature/Model Extraction (training data)**
   With a certain regularized parameter λ, solve the ℓ_1 regularized regression problem (Eq. (26) for nonlinear inelastic or Eq. (8) for nonlinear elastic) by LASSO to get sparse model parameters and its corresponding AIC value.
5. **Model Selection**
   Repeat step 2 to step 4 using various values of λ and form an AIC curve. Pick the optimal λ where the AIC curve has significant slope change.
6. **Sparse Feature/Model Evaluation (testing data)** Testing ground motion as input to the sparse model to verify the generalization ability of the model. Note: testing data is not used in training as dictated by cross validation.

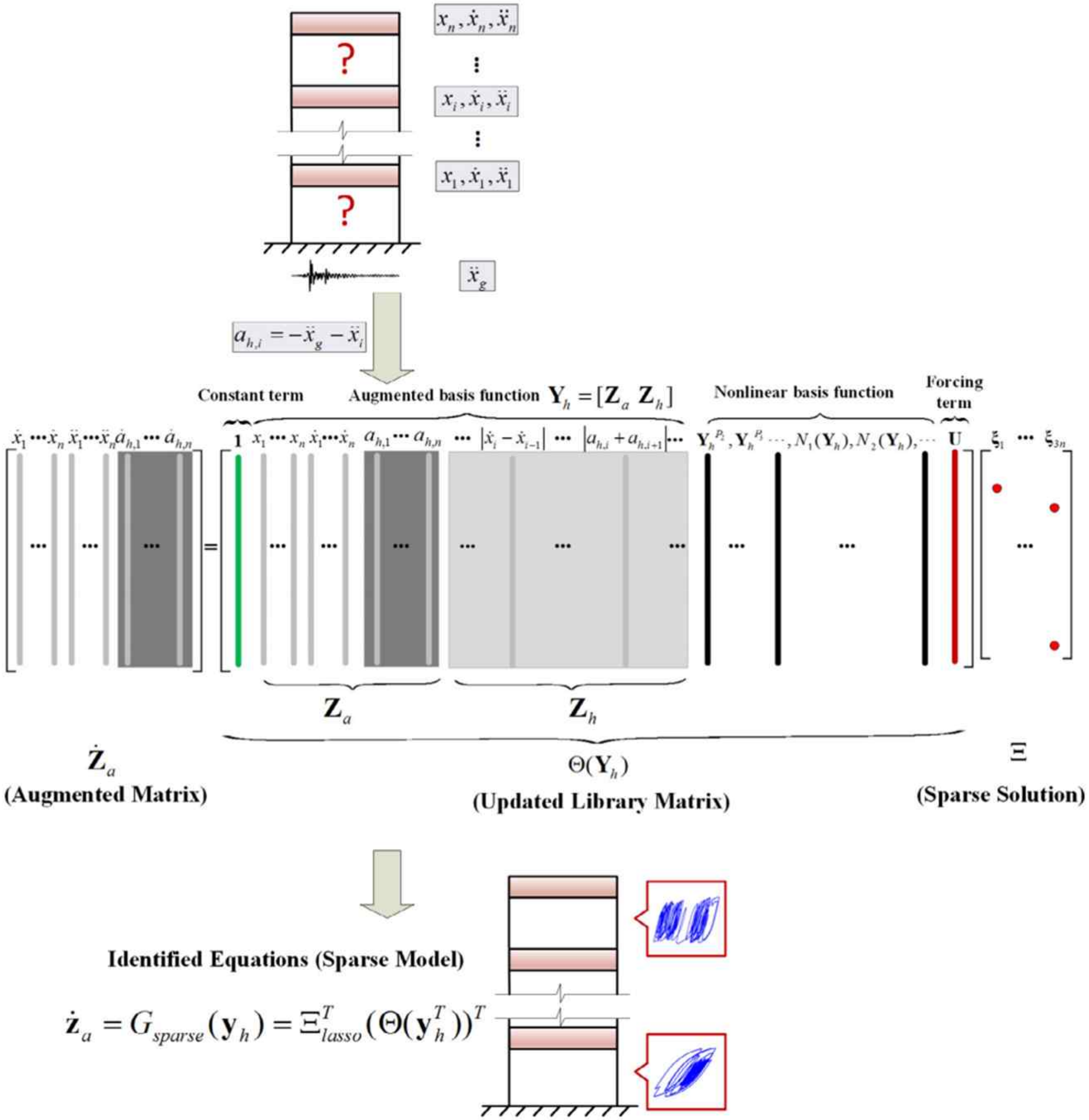


Fig. 3. Schematic procedure of proposed augmented sparse identification for nonlinear inelastic structural systems with hysteresis and permanent deformation.

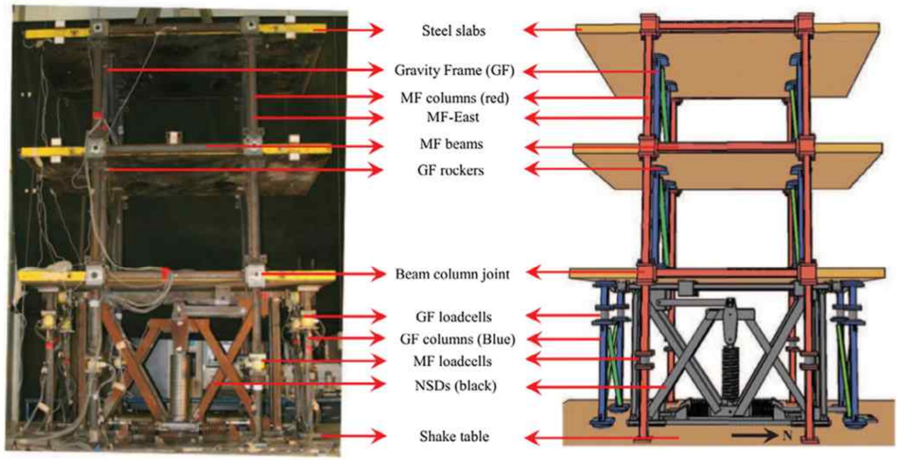


Fig. 18. Shake table testing of a 3-story structure with a NSD installed in the first floor.

Comparison

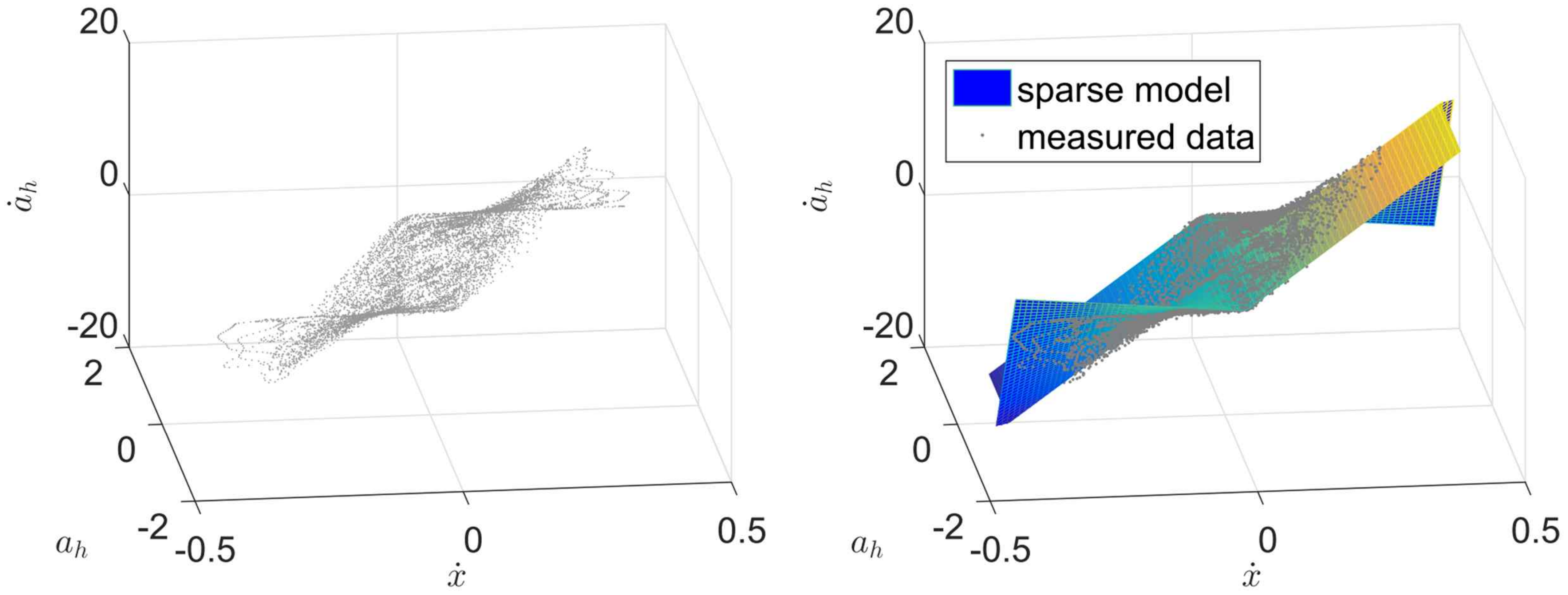
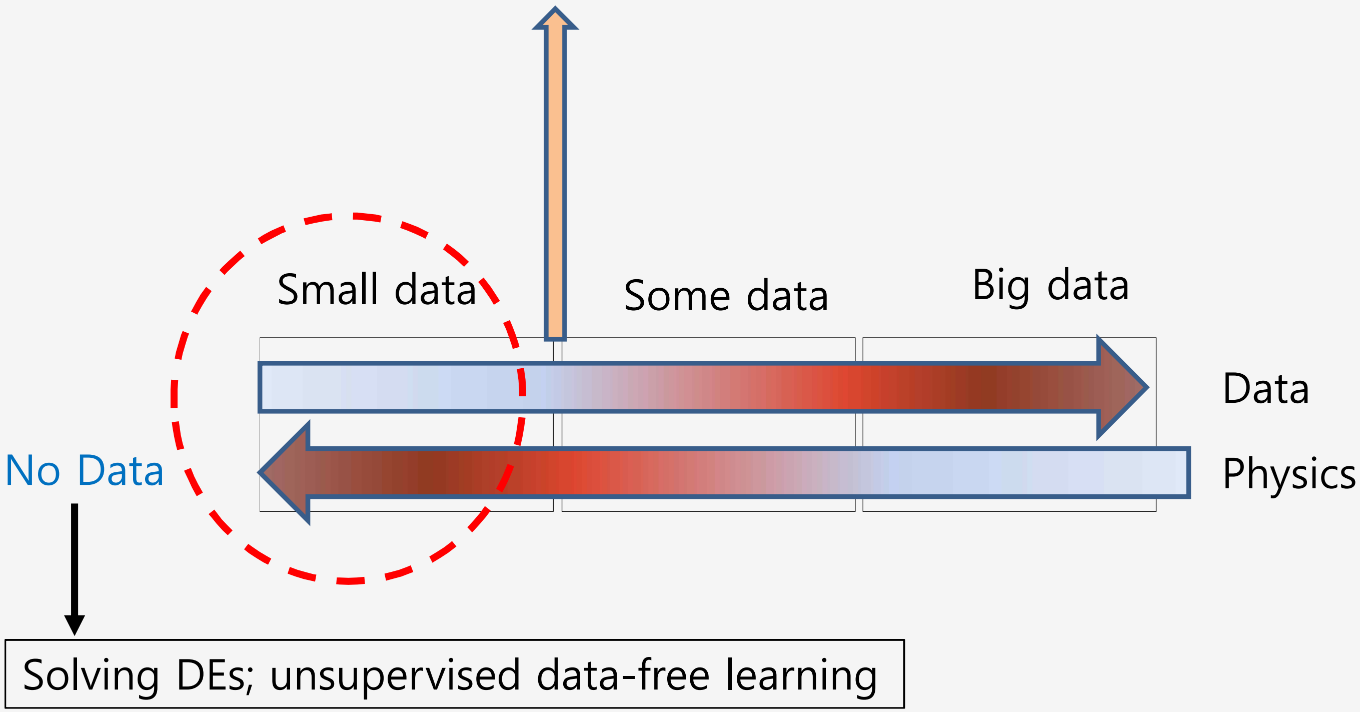


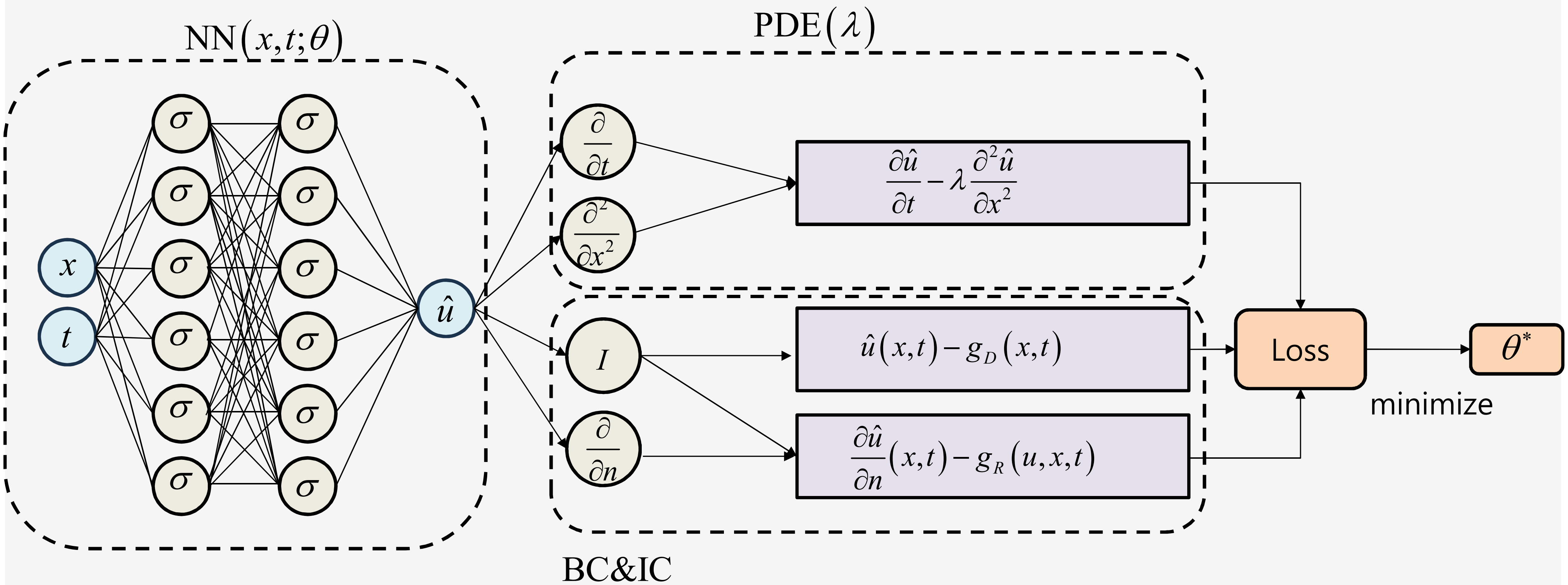
Fig. 11. measured data (left); comparison between measured data and function surface identified in the sparse model (right).

Physics informed Neural Network



Why SciML

- Scientific machine learning is an emerging discipline within the data science community. SciML seeks to address domain-specific data challenges and extract insights from scientific data sets through innovative methodological solution.
- SciML draws on tools from both machine learning and scientific computing to develop new methods for scalable, domain-aware, robust, reliable, and interpretable learning and data analysis, and will be critical in driving **the next wave** of **data-driven scientific discovery** in the physical and engineering sciences.



$$\frac{\partial \hat{u}}{\partial t} - \lambda \frac{\partial^2 \hat{u}}{\partial x^2}$$

Lu et al 2019

Behavior of system and Optimization objective

Partial differential equation: $F(u;\theta)(\mathbf{x})=0$ → $F(u;\theta)(\mathbf{x})=F(u,\mathbf{x};\theta)=0, \quad x\in\Omega$

Initial conditions: $I(u;\theta)(x,t_0)=0$

Boundary conditions: $B(u;\theta)(x,t)=0$

Optimization objective: $\min_{f\in H} L(f;D)+\Omega(g)$

$$L=\frac{\lambda_r}{N_r}\sum_{i=1}^{N_r}\left\|F(u_w;\theta)(\mathbf{x}_i)\right\|^2+\frac{\lambda_i}{N_i}\sum_{i=1}^{N_i}\left\|I(u_w;\theta)(\mathbf{x}_i)\right\|^2$$

Residual loss — Initial condition

$$+\frac{\lambda_b}{N_b}\sum_{i=1}^{N_b}\left\|B(u_w;\theta)(\mathbf{x}_i)\right\|^2+\frac{\lambda_d}{N_d}\sum_{i=1}^{N_d}\left\|u_w(\mathbf{x}_i)-u(\mathbf{x}_i)\right\|^2$$

Boundary condition — Regular data loss

Loss Reweighting

$$\hat{\lambda}_i = \frac{\max\left\{\nabla_w L_r\left(w_n\right)\right\}}{\left|\nabla_w L_r\left(w_n\right)\right|}$$

$$K = \begin{pmatrix} K_{bb} & K_{br} \\ K_{br} & K_{rr} \end{pmatrix}$$

$$\left(K_{bb}\right)_{i,j} = \left\langle \frac{du_w\left(\mathbf{x}_i^b\right)}{dw}, \frac{du_w\left(\mathbf{x}_j^b\right)}{dw} \right\rangle$$

$$\left(K_{bb}\right)_{i,j} = \left\langle \frac{du_w\left(\mathbf{x}_i^b\right)}{dw}, \frac{dF\left(u_w\right)\left(\mathbf{x}_j\right)}{dw} \right\rangle$$

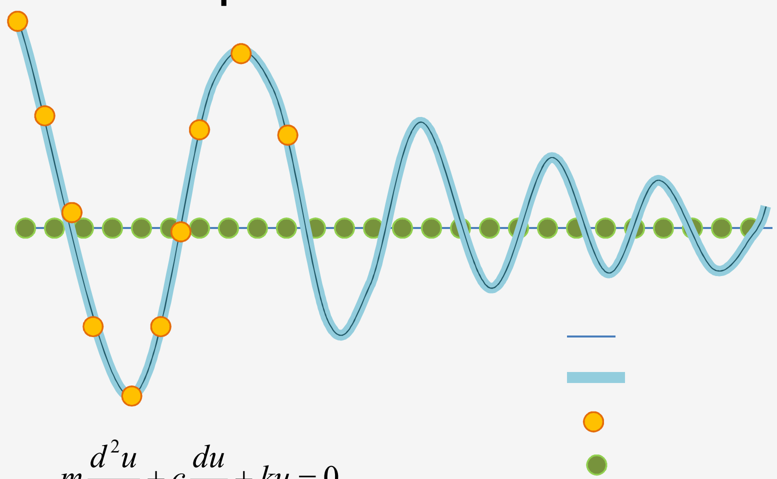
$$\left(K_{bb}\right)_{i,j} = \left\langle \frac{dF\left(u_w\right)\left(\mathbf{x}_i\right)}{dw}, \frac{dF\left(u_w\right)\left(\mathbf{x}_j\right)}{dw} \right\rangle$$

$$\lambda_b = \frac{Tr\left(K\right)}{Tr\left(K_{bb}\right)}$$

$$\lambda_r = \frac{Tr\left(K\right)}{Tr\left(K_{rr}\right)}$$

Advantage and limitation

- Example



$$m\frac{d^2u}{dt^2}+c\frac{du}{dt}+ku=0$$

$$NN(t;\theta)\approx u(t)$$

Boundary loss

$$L(\theta)=\frac{1}{N}\sum_i^N\left(NN(t_i;\theta)-u(t_i)\right)^2$$

$$+\frac{\lambda}{M}\sum_j^M\left(\left[m\frac{d^2}{dt^2}+c\frac{d}{dt}+k\right]NN(t_j;\theta)\right)^2$$

Physics loss

Advantages

- Mesh-free
- Can jointly sole forward and inverse problems
- Mostly unsupervised
- Can perform well for high-dimensional PDEs

Limitation

- Computational cost often high
- Can be hard to optimize
- Challenging to scale to high-frequency, multi-scale problems

from ETH 401-4656-21L Deep learning in Scientific computing 2023

Neural operator

DeepONets
Physical Informed DeepONets
Graph Operator Networks
Fourier Operator Networks
PINO

Caltech group
Nikola Kovachki
Zongyi Li
Anima Anandkumar
Andrew Stuart
....

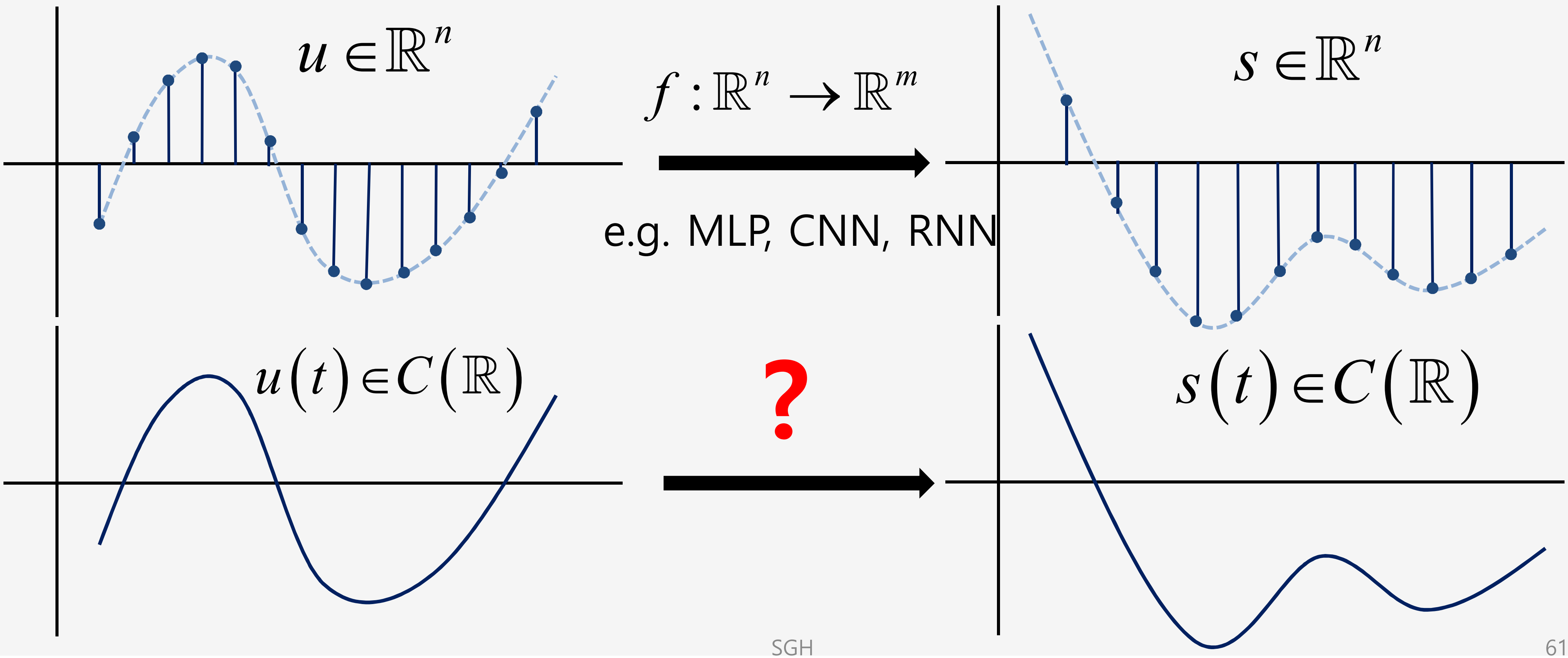
Brown University
Lu Lu
Pengzhan Jin
George Em Karniadakis

Not vector-to-vector mapping

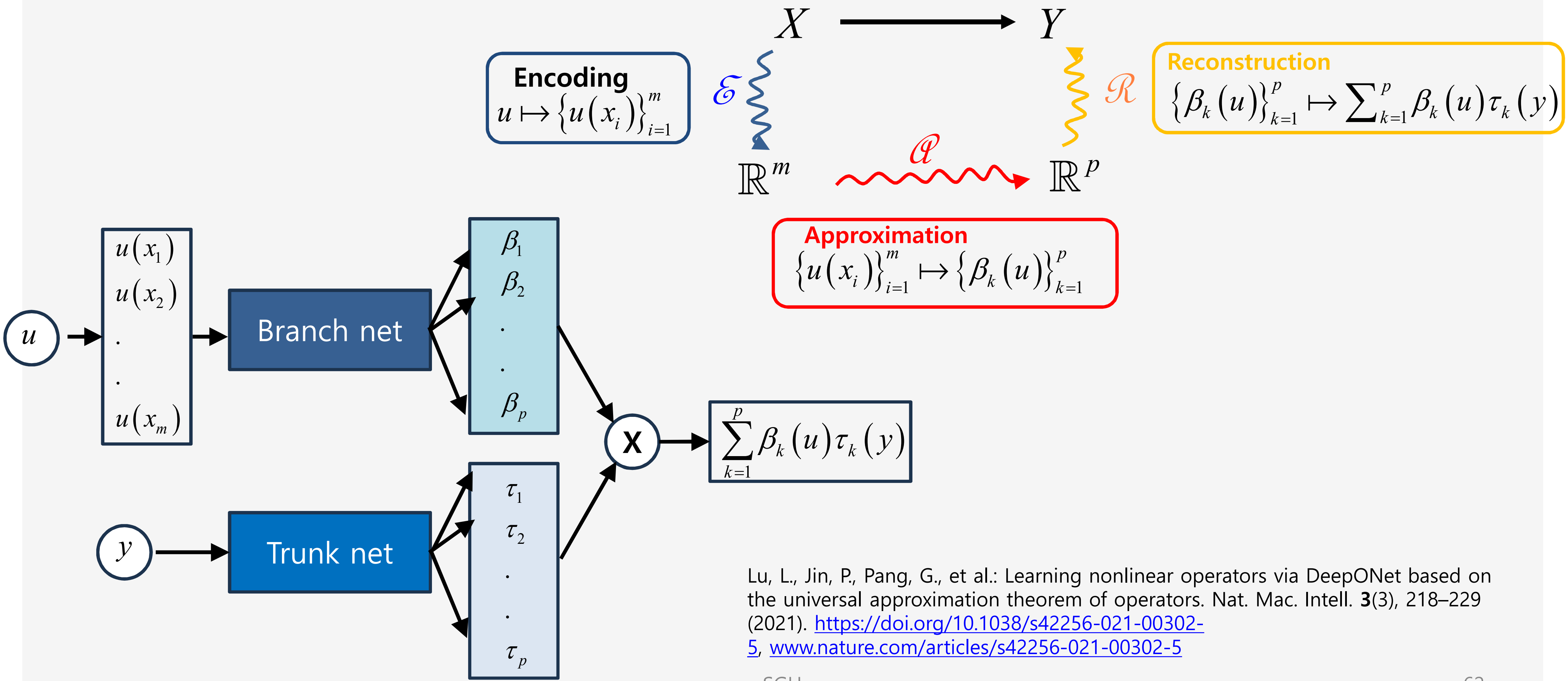
Function-to-function mapping

How to learn function space

Take discrete measurement of functional data and use standard ML models on the finite dimensional discretization



DeepOperatorNet



Lu, L., Jin, P., Pang, G., et al.: Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nat. Mac. Intell. **3**(3), 218–229 (2021). https://doi.org/10.1038/s42256-021-00302-5, www.nature.com/articles/s42256-021-00302-5

Goal of Neural Operator

To approximate a latent operator, i.e.,
a mapping between parameters and the state variable

$$\min_{w \in W} \left\| G_w(\theta)(\mathbf{x}) - \tilde{G}(\theta)(\mathbf{x}) \right\|$$

DeepONets

$$G_w(\theta)(\mathbf{x}) = b_0 + \sum_{k=1}^{p} b_k(\theta) t_k(\mathbf{x})$$

Trunk network

Branch network

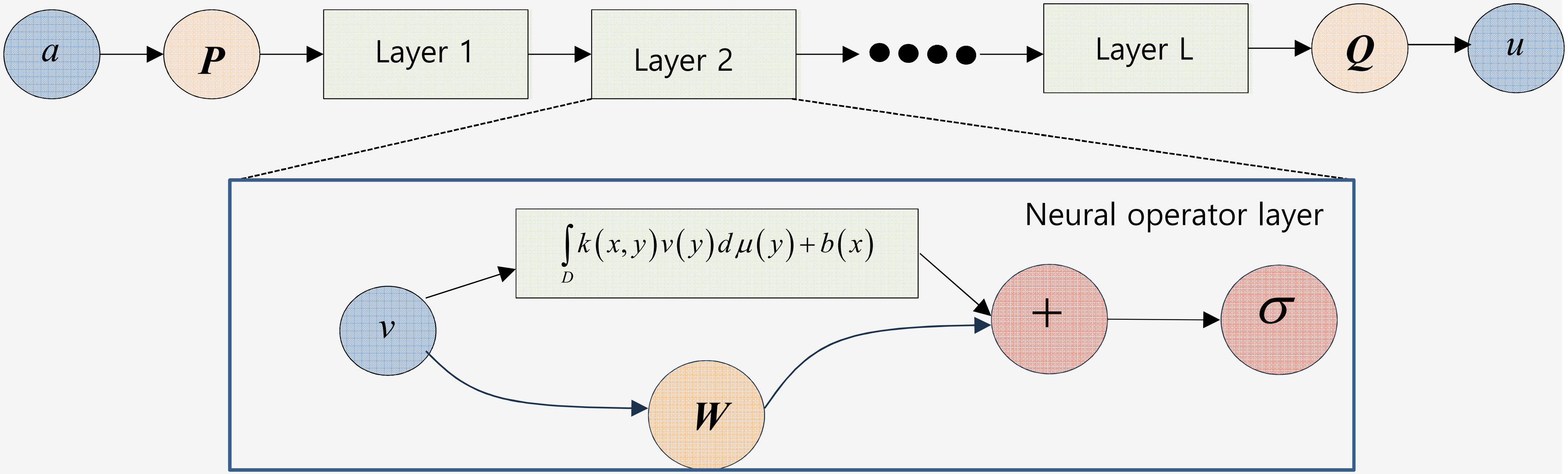
Neural operator

$$u = Q\left(K_l \circ \sigma_l \circ \cdots \sigma_1 \circ K_0\right) P v$$

P, Q are local network (encoder, decoder)

P lifts the input to a high dimensional channel space.
Q projects the representation back to the original space.

Neural operator architecture schematic



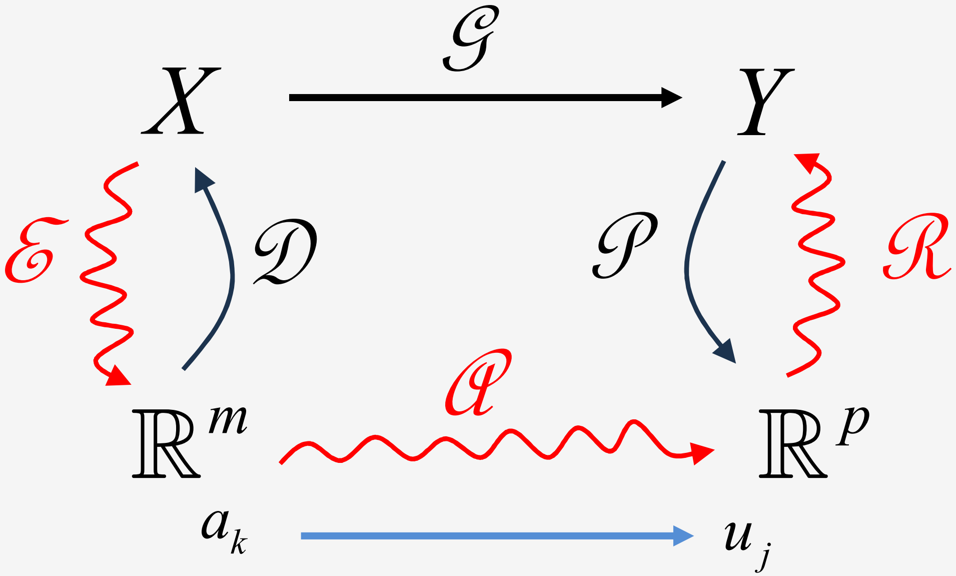
The input function *a* is passed to a pointwise lifting operator ***P*** that is followed by ***T*** layers of integral operators and pointwise non-linearity operator σ. In the end, the pointwise projection operator ***Q*** outputs the function *u*.

Operator Learn Architecture

$a \in X$

$a \approx \sum_{k=1}^{m} a_k \varphi_k(x)$

$\mathcal{E}\, a = \{a_k\}_{k=1}^{m}$



$\mathcal{R}(\{u_j\}) \rightarrow \sum_{j=1}^{p} u_j \varphi_j(x)$

Architecture	Encoder	Approximator	Reconstructor
Spectral Neural Operator	Coeffs	DNNs	Fourier/Chebychev basis

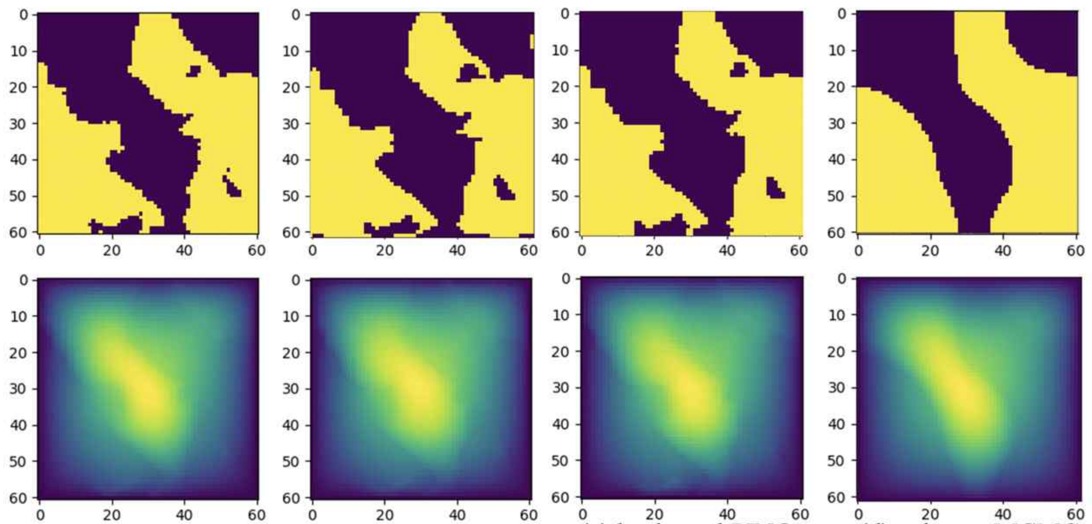
2D steady Darcy flow equation

$$-\nabla\cdot\left(a(x)\nabla u(x)\right)=f(x) \qquad x\in(0,1)^2$$

$$u(x)=0 \qquad x\in\partial(0,1)^2$$

PHYSICS-INFORMED NEURAL OPERATOR FOR LEARNING PARTIAL DIFFERENTIAL EQUATIONS
By Anonymous authors

Under review as a conference paper at ICLR 2022



(a) ground truth (b) forward PINO (c) backward PINO (d) solver + MCMC

Figure 3: Darcy inverse problem

Fourier Layer

Use convolution as integral operator
and implement with Fourier transform

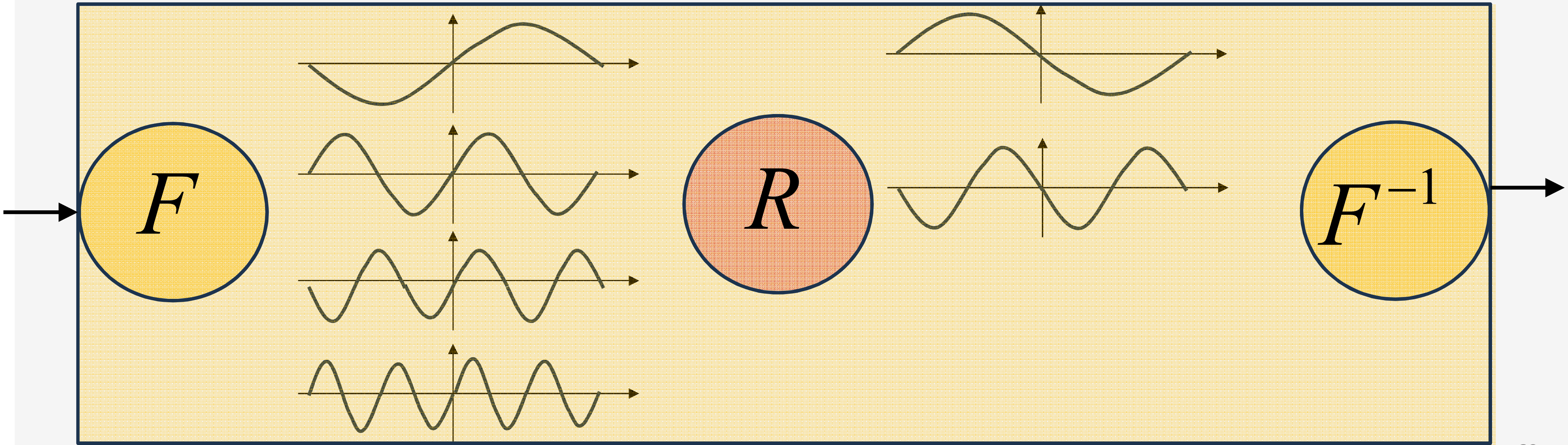
$$\left(K\left(a;\phi\right)v_t\right) := \int_D k\left(x, y, a\left(x\right), a\left(y\right);\phi\right)v_t\left(y\right)dy$$

$$\left(K\left(\phi\right)v_t\right) = F^{-1}\left(R_\phi \bullet \left(Fv_t\right)\right)\left(x\right)$$

Fourier layer

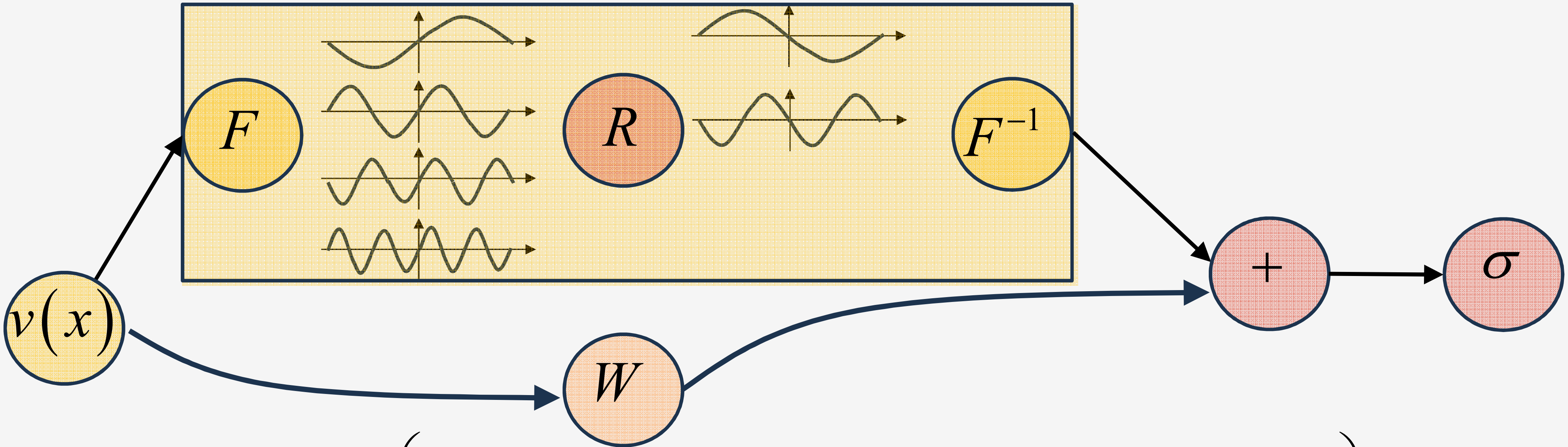
1. Fourier transform
2. Linear transform
3. Inverse Fourier transform

$$\left(K(\phi)v_t\right) = F^{-1}\left(R_\phi \cdot \left(Fv_t\right)\right)(x)$$



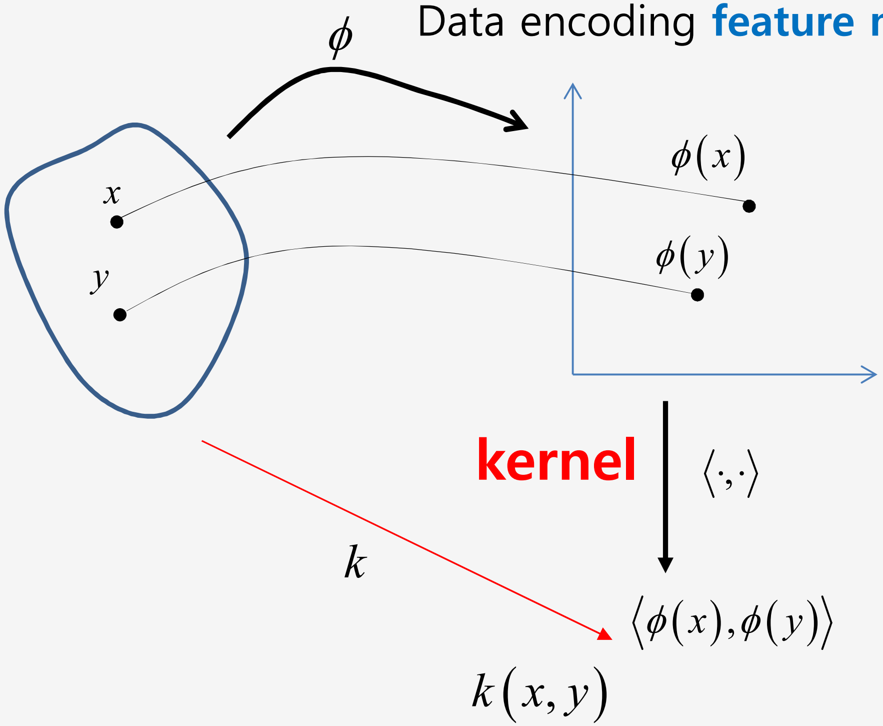
Fourier layer

The linear transform W outside keep the track of the location information (x) and non-periodic boundary



$$v_{t+1}(x) = \sigma\left(W v_t(x) + \int_D k_\phi(x, y, a(x), a(y)) v_t(y) v_x(dy) \right)$$

Kernelization



Polynomial kernel example

$$\phi : \mathbb{R}^2 \Rightarrow \mathbb{R}^3$$

$$\phi : x_1, x_2 \rightarrow z_1, z_2, z_3$$

$$\text{where} \quad z_1 = \sqrt{2} x_1 x_2 \quad z_2 = x_1^{\ 2} \quad z_3 = x_2^2$$

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle$$

$$k(\mathbf{x}, \mathbf{z}) = \left(\mathbf{x}^{\mathrm{T}} \mathbf{z}\right)^2 = (x_1 z_1 + x_2 z_2)^2$$

$$= x_1^2 z_1^2 + 2 x_1 z_1 x_2 z_2 + x_2^2 z_2^2$$

$$= \left(x_1^2, \sqrt{2} x_1 x_2, x_2^2\right) \begin{pmatrix} z_1^2 \\ \sqrt{2} z_1 z_2 \\ z_2^2 \end{pmatrix}$$

$$= \phi(\mathbf{x})^T \phi(\mathbf{z})$$

Kernel Trick

- Computation in explicit, **high-dimensional feature maps are expensive**
- For some feature maps, we can, however, compute distances between point cheaply
  - Without explicitly constructing the high-dimensional space at all
- Example: quadratic feature map for $\mathbf{x} = (x_1, \cdots, x_p)$ ⇨

$p\_\text{dimension} \Rightarrow 3p\_\text{dimension}$

$$\Phi(\mathbf{x}) = \left(x_1, \cdots, x_p, x_1^2, \cdots, x_p^2, \sqrt{2}x_1x_2, \cdots, \sqrt{2}x_{p-1}x_p\right)$$

- **A kernel function exists** for this feature map to compute dot products

$$k_{quad}(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j + (\mathbf{x}_i \cdot \mathbf{x}_j)^2$$

- **Skip computation of** $\Phi(\mathbf{x}_i)$ and $\Phi(\mathbf{x}_j)$ and compute $k(\mathbf{x}_i, \mathbf{x}_j)$ **directly**

Kernel Function $k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$

"Gram Matrix"

$$K = \begin{bmatrix} \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_1) & \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2) & \cdot & \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_N) \\ \phi(\mathbf{x}_2)^T \phi(\mathbf{x}_1) & \phi(\mathbf{x}_2)^T \phi(\mathbf{x}_2) & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_1) & \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_2) & \cdot & \phi(\mathbf{x}_N)^T \phi(\mathbf{x}_N) \end{bmatrix}$$

$$= \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \bullet & k(\mathbf{x}_1, \mathbf{x}_N) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \bullet & \bullet \\ \cdot & \cdot & \bullet & \bullet \\ k(\mathbf{x}_N, \mathbf{x}_1) & k(\mathbf{x}_N, \mathbf{x}_2) & \bullet & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

New data

$$f(x) = \sum_t w_t \phi(\mathbf{x}_t) \cdot \phi(\mathbf{x})$$

$k(\mathbf{x}_t, \mathbf{x})$

training data

New data

$$\Psi_{GPS}(\mathbf{x} : p, \theta) = \exp\left(\sum_t^{N_t} w_t k_p^\theta (\mathbf{x}, \mathbf{x}_t) \right)$$

kernel

Training data

Exponential ensures **product** separable

Noiseless GP regression

We observe a training set $D = \{(\mathbf{x}_i \mathbf{f}_i), i = 1:N\}$ where $f_i = f(\mathbf{x}_i)$

Given a test set X* of size N*x D, we want to **predict the function output** f*.

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(\begin{pmatrix} \mu \\ \mu_* \end{pmatrix}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right)$$

$$k(x,x') = \sigma_f^2 \exp\left(-\frac{1}{2l^2}(x-x')^2 \right)$$

$K(X,X): N \times N$

$K(X,X_*): N \times N_*$

$K(X_*,X_*): N_* \times N_*$

$$p(\mathbf{f}_* | \mathbf{X}_*, \mathbf{X}, \mathbf{f}) = N(\mathbf{f}_* | \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

$$\boldsymbol{\mu}_* = \boldsymbol{\mu}(\mathbf{X}_*) + \mathbf{K}_*^T \mathbf{K}^{-1}(\mathbf{f} - \boldsymbol{\mu}(\mathbf{X}))$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{K}_*$$

Noisy GP regression

$$y = f(X) + \varepsilon, \quad \text{where } \varepsilon \sim N\left(0, \sigma_y^2\right)$$

$$p(\mathbf{y}|\mathbf{X}) = \int p(\mathbf{y}|\mathbf{f}, \mathbf{X}) p(\mathbf{f}|\mathbf{X}) d\mathbf{f}$$

$$p(\mathbf{f}|\mathbf{X}) = N(\mathbf{f}|\mathbf{0}, \mathbf{K})$$

$$p(\mathbf{y}|\mathbf{f}) = \Pi_i N\left(y_i | f_i, \sigma_y^2\right)$$

$$\operatorname{cov}[\mathbf{y}|\mathbf{X}] = \mathbf{K} + \sigma_y^2 \mathbf{I}_N \triangleq \mathbf{K}_y$$

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K_y & K_* \\ K_*^T & K_{**} \end{bmatrix}\right)$$

$$p(\mathbf{f}_* | \mathbf{X}_*, \mathbf{X}, \mathbf{f}) = N(\mathbf{f}_* | \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

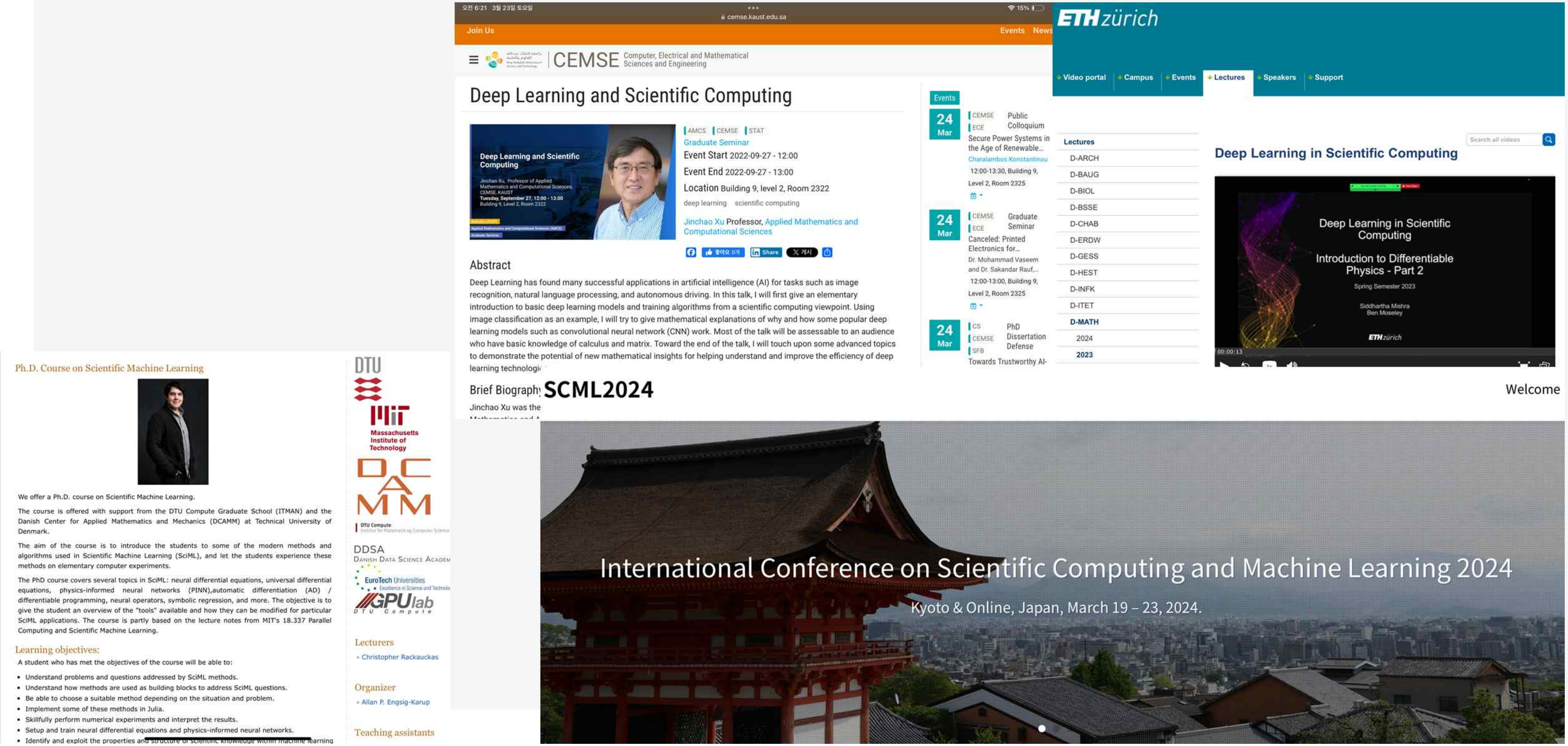
$$\boldsymbol{\mu}_* = \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{y}$$

$$\boldsymbol{\Sigma}_* = \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{K}_*$$

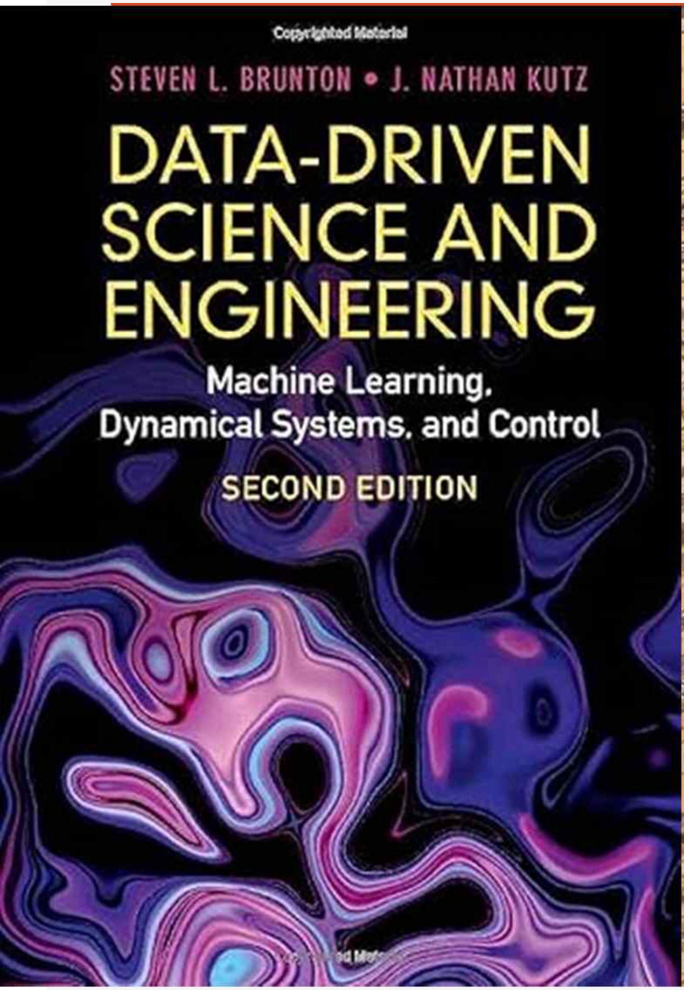
Concluding remark: Takeaway (by Zongyi Li @caltech)

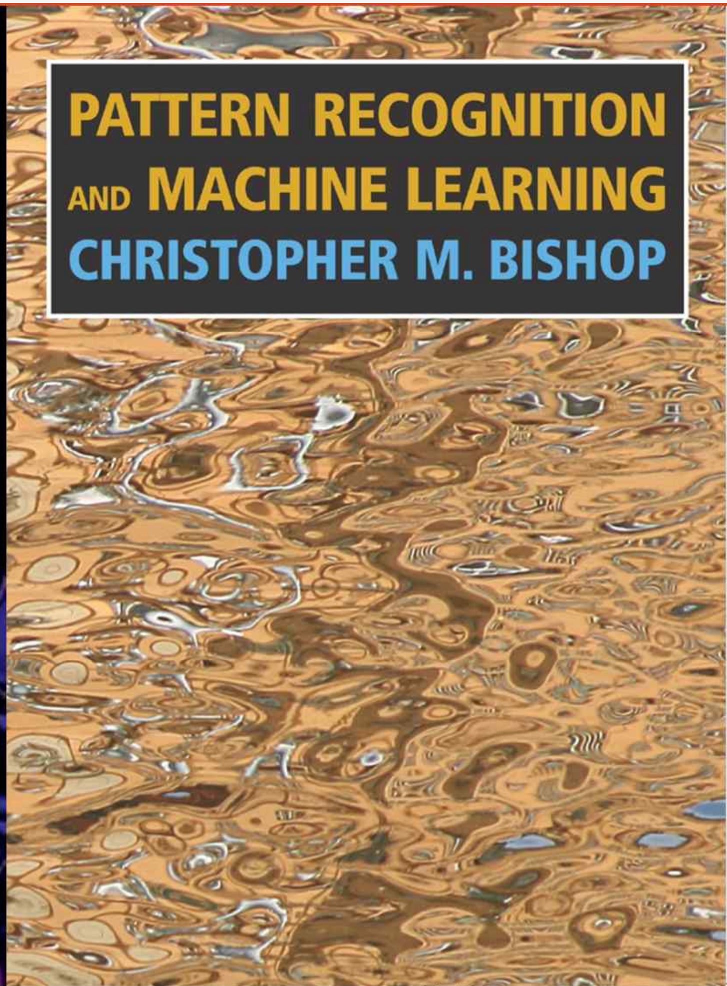
1. Data-driven ML: learn the equation
2. Neural operator-learning: parametrize the mesh-invariant operator
3. Fourier method: efficient for continuous inputs and outputs
4. SciML: accurate than other deep learning method, faster than conventional solvers
5. Future: scale up for engineering applications

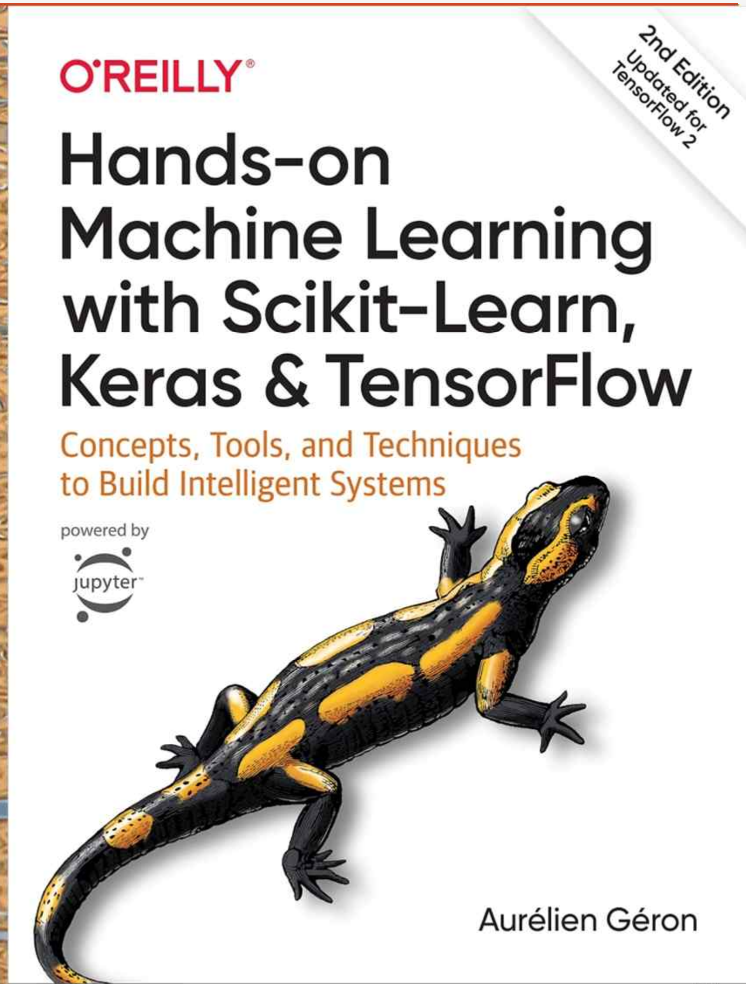
Education and conference



References


Copyrighted Material
STEVEN L. BRUNTON • J. NATHAN KUTZ
DATA-DRIVEN SCIENCE AND ENGINEERING
Machine Learning, Dynamical Systems, and Control
SECOND EDITION


PATTERN RECOGNITION AND MACHINE LEARNING
CHRISTOPHER M. BISHOP

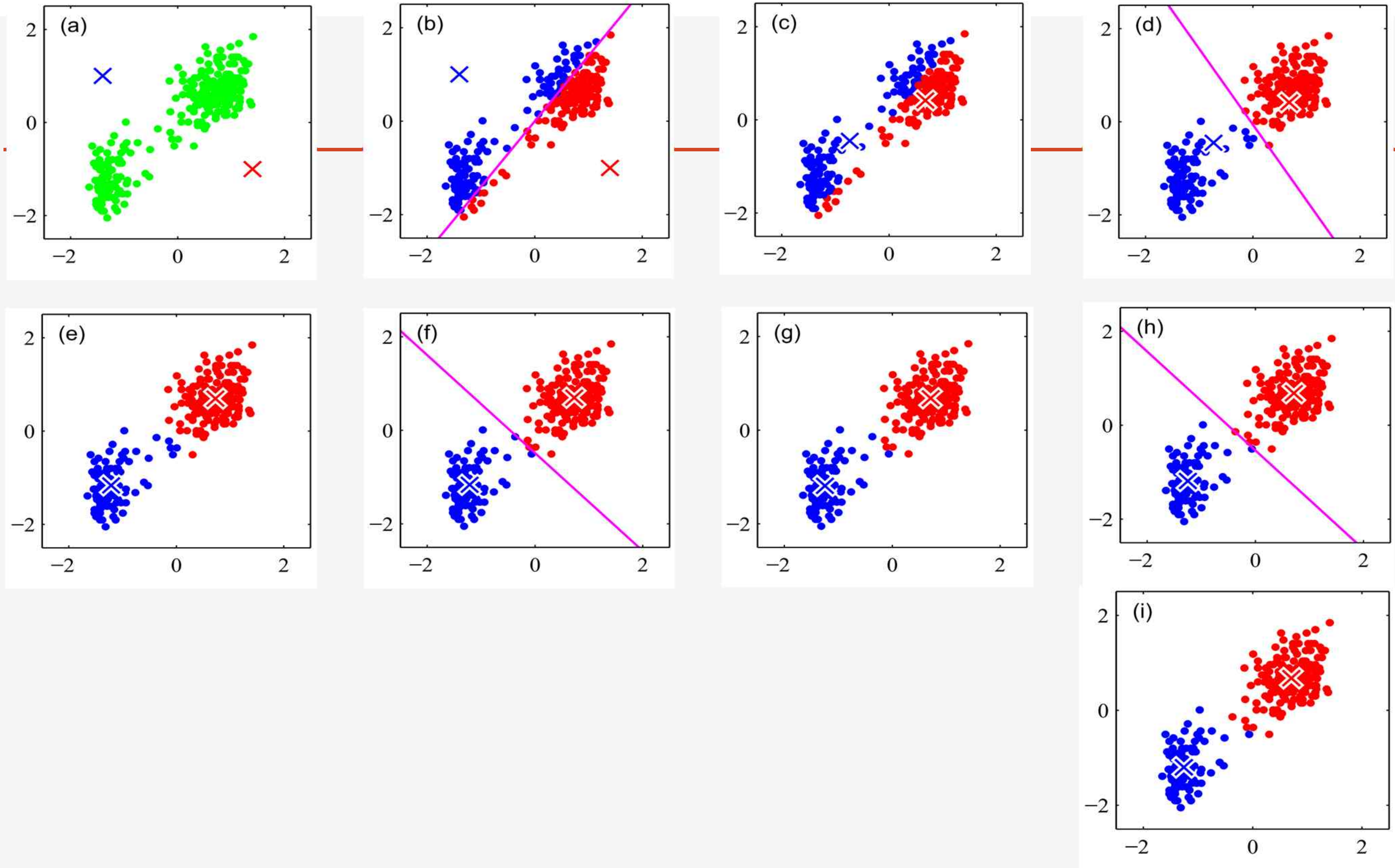

O'REILLY
2nd Edition
Updated for TensorFlow 2
Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow
Concepts, Tools, and Techniques to Build Intelligent Systems
powered by
jupyter
Aurélien Géron

Thank you

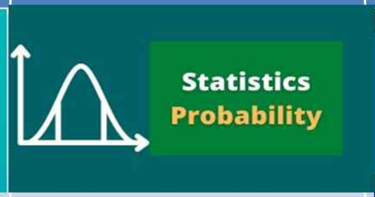
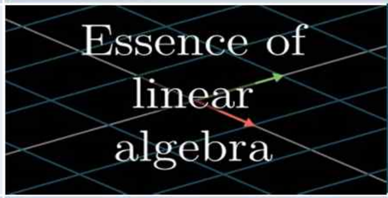
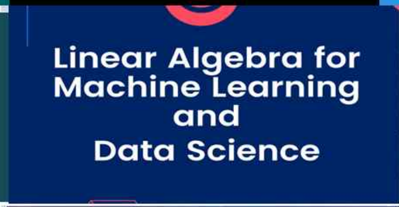
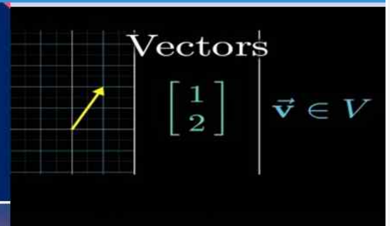
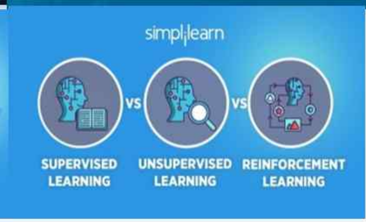
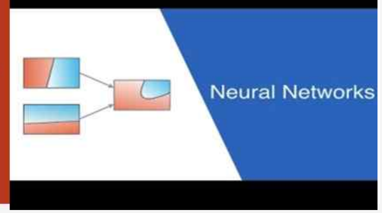
Prerequisite

- Linear Algebra
- Probability
- Data Science
- Information Theory
- Programing
- Coding (Python.....)
- **Problem Setting**

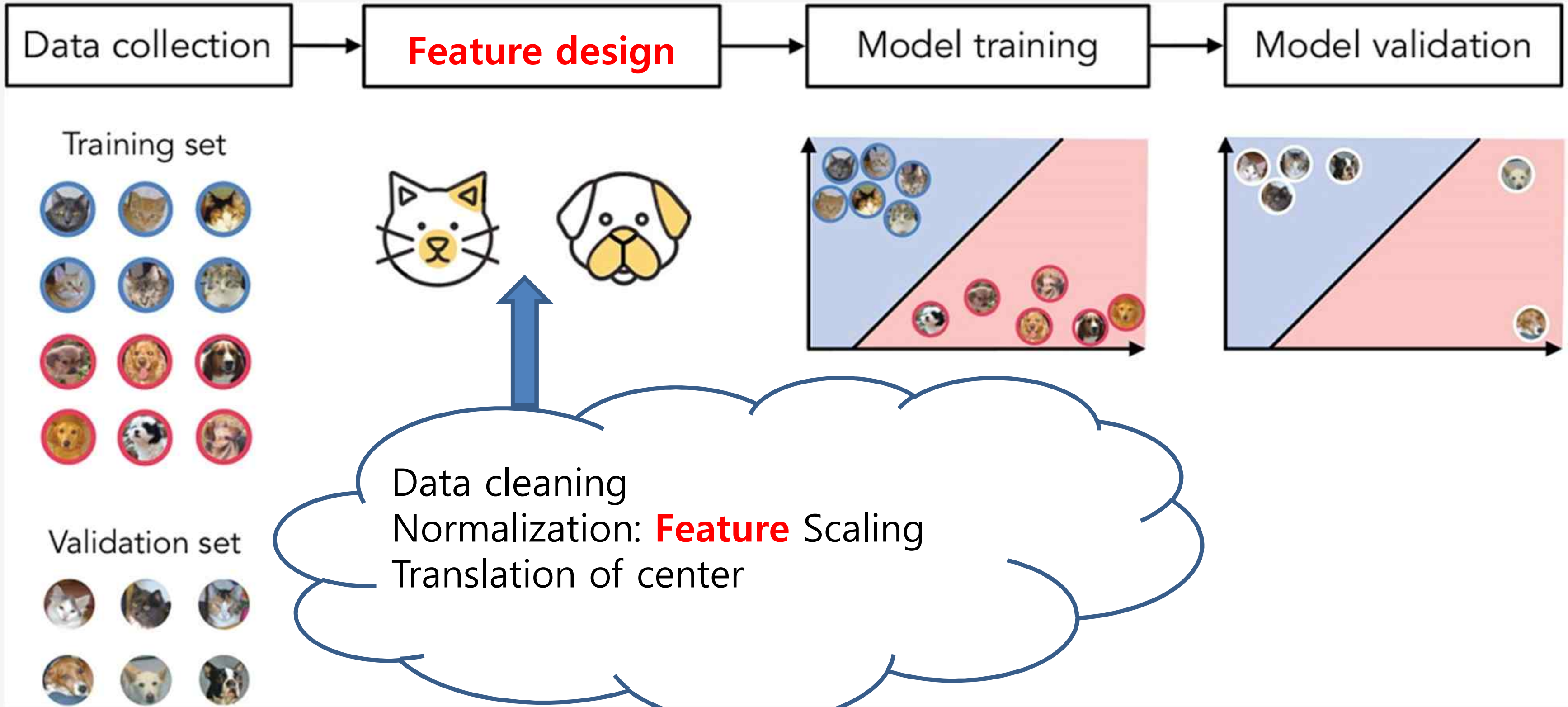

(a)
2
0
−2
−2
0
2


(a)
(b)
(c)
(d)
(e)
(f)
(g)
(h)
(i)
2
0
−2
−2
0
2

Youtube 학습

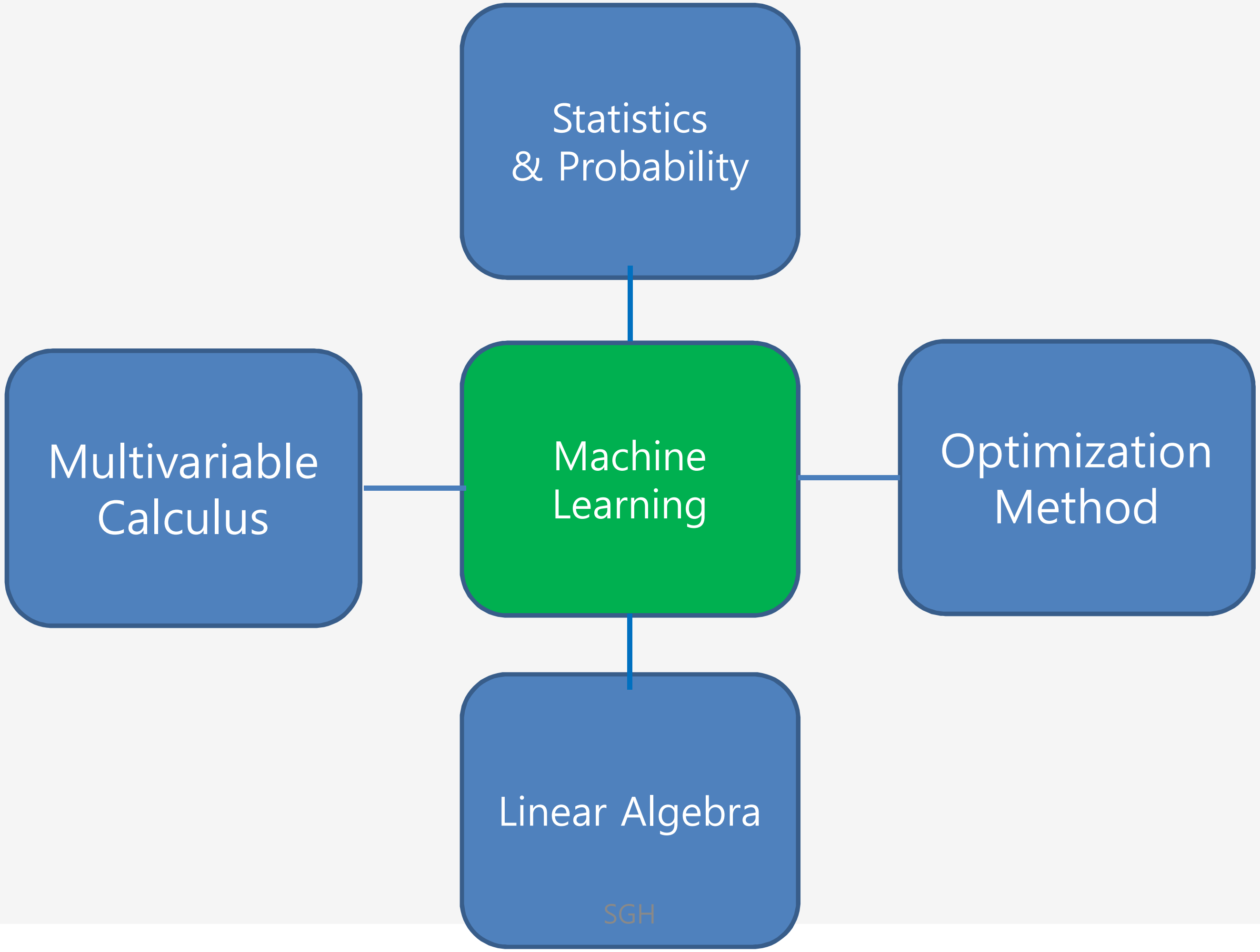
Subject	1	2	3	4
probability				
Linear algebra				
Data prep.				
ML algorithm				

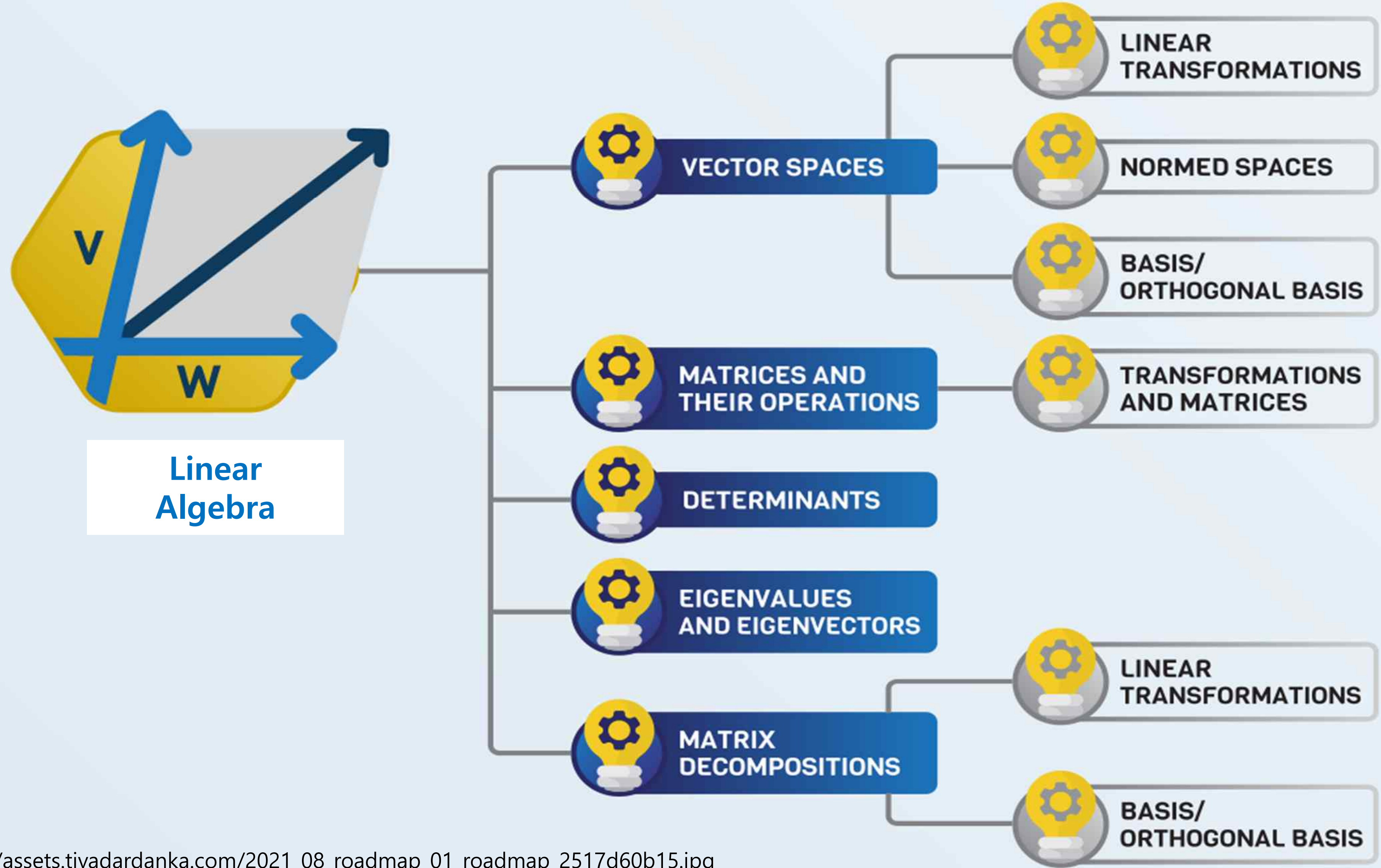
ML Process



Data cleaning
Normalization: **Feature** Scaling
Translation of center

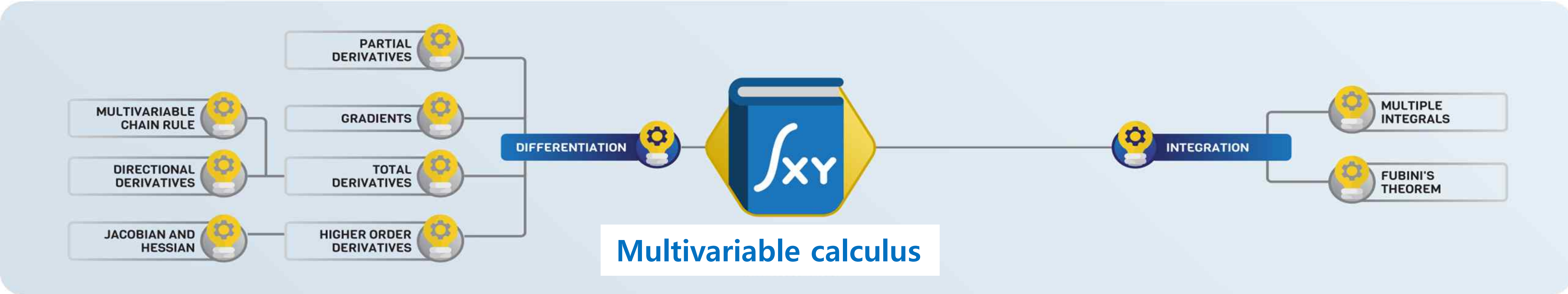
Math Skills for Machine Learning





https://assets.tivadardanka.com/2021_08_roadmap_01_roadmap_2517d60b15.jpg

Composite function의 이해



Regularization의 의미와 활용
Backpropagation
Jacobian

Magic
of ML and DL

NEURAL NETWORKS