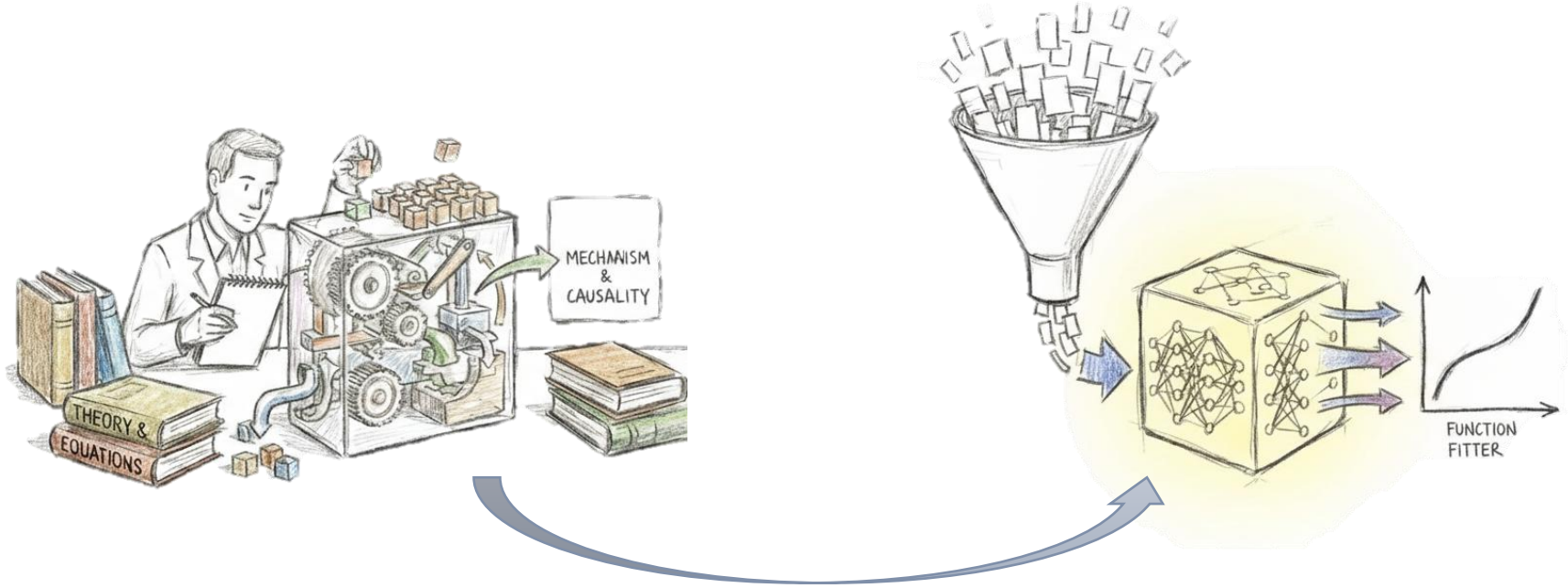
物理情報を活用した機械学習によるモデリング

武石 直也（東京大学）

2026-03-03



なぜ物理情報を活用したい？

- ~~気持ちの問題~~
- 必要な訓練データ量を抑える
- データを取得できない領域での挙動
- 物理的におかしくない予測

講演の内容

- **物理学的な事前知識**を活用しつつ**機械学習**でデータに基づいてモデリングを行う方法について、方法論の立場から概観する
  - 物理知識に基づく **① 訓練データ、② モデル、③ 目的関数** の設計

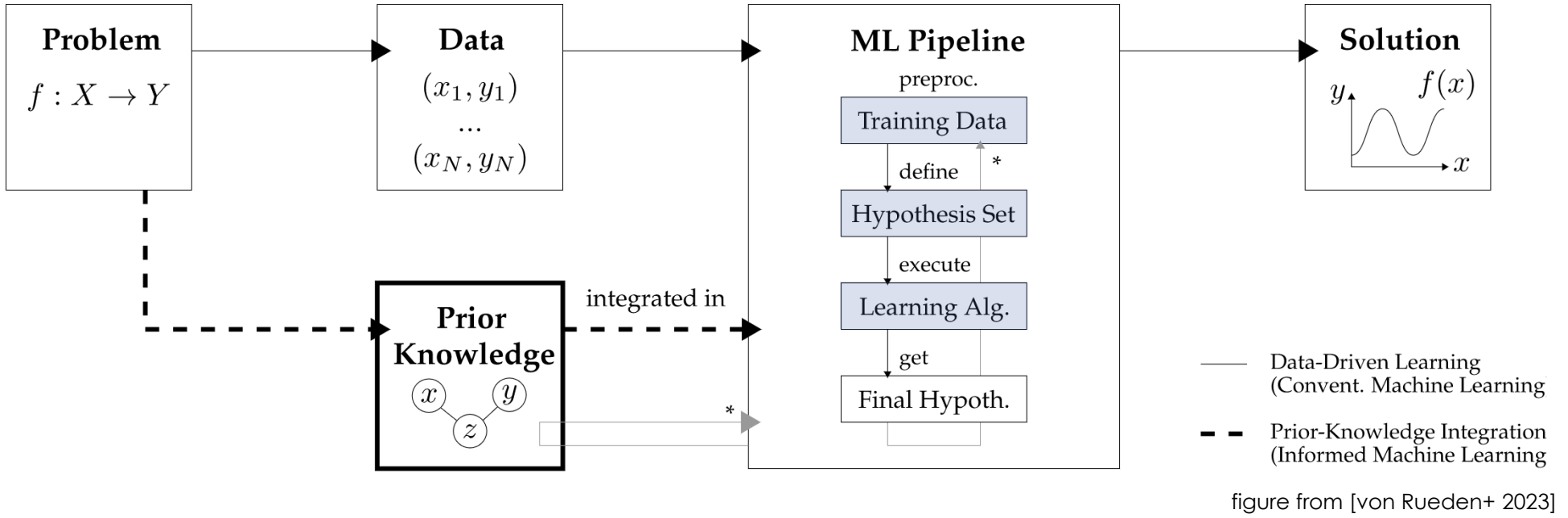


figure from [von Rueden+ 2023]

von Rueden+, Informed Machine Learning – A Taxonomy and Survey of Integrating Knowledge into Learning Systems, *IEEE Transactions on Knowledge and Data Engineering*, 2023

物理情報に基づく目的関数設計

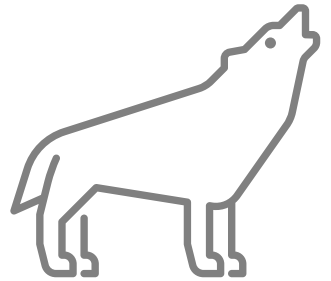
物理情報に基づく目的関数

- データ$\{(x_1, y_1), \ldots, (x_n, y_n)\}$と 物理情報による制約 $R(x, y) = 0$ の両方を満たすように学習する

$$\text{minimize} \sum L(x, y) + \lambda R(x, y)$$

二乗誤差など

- 厳密に $R(x, y) = 0$ を要請せず、ソフト制約として扱うことが多い
- 例
  - physics-informed neural nets (PINNs)
  - 制約付き生成モデル



Physics-informed neural nets

- 偏微分方程式

$$\frac{\partial u(t,\boldsymbol{x})}{\partial t} - Lu(t,\boldsymbol{x}) = 0 \quad \text{for} \quad t \in [0,T],\ \boldsymbol{x} \in \Omega$$

- 初期条件、境界条件

$$u(t_0,\boldsymbol{x}) = u_0(\boldsymbol{x}) \quad \text{for} \quad \boldsymbol{x} \in \Omega$$
$$u(t,\boldsymbol{x}) = g(t,\boldsymbol{x}) \quad \text{for} \quad t \in [0,T],\ \boldsymbol{x} \in \partial\Omega$$

- 深層ニューラルネット $u_\theta(t,\boldsymbol{x})$ で解 u を表す [Raissi+ 2019; Sirignano+ 2018]
  - メッシュフリーな表現ができる
  - 微分方程式とデータの両方から情報を使える

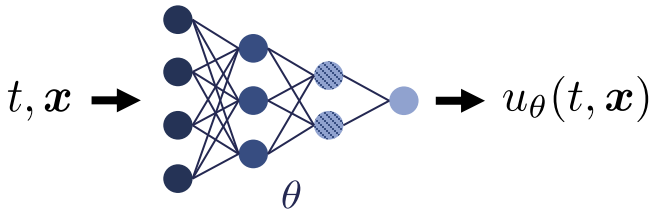
Raissi+, Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations, *J. Computational Physics*, 2019

Sirignano and Spiliopoulos, DGM: A Deep Learning Algorithm for Solving Partial Differential Equations, *J. Computational Physics*, 2018

Physics-informed neural nets

- 深層ニューラルネット $u_\theta(t, \boldsymbol{x})$ で解 u を表す [Raissi+ 2019; Sirignano+ 2018]

学習すべきパラメタ



  - 目的関数:

$$
\begin{aligned}
\underset{\theta}{\text{minimize}}\ \lambda_{\text{eq}} & \sum_{t,\boldsymbol{x}\in[0,T]\times\Omega} \left\| \frac{\partial u_\theta(t,\boldsymbol{x})}{\partial t} - L u_\theta(t,\boldsymbol{x}) \right\|^2 && \text{… 微分方程式の残差} \\
+ \lambda_{\text{ic}} & \sum_{\boldsymbol{x}\in\Omega} \left\| u_\theta(t_0,\boldsymbol{x}) - u_0(\boldsymbol{x}) \right\|^2 && \text{… 初期条件} \\
+ \lambda_{\text{bc}} & \sum_{t,\boldsymbol{x}\in[0,T]\times\partial\Omega} \left\| u_\theta(t,\boldsymbol{x}) - g(t,\boldsymbol{x}) \right\|^2 && \text{… 境界条件} \\
+ \lambda_{\text{data}} & \sum_{(t,\boldsymbol{x},u)\in S} \left\| u_\theta(t,\boldsymbol{x}) - u \right\|^2 && \text{… データ}
\end{aligned}
$$

PINNsの課題

- 最適化手法
  - lossを下げきるにはsecond-order
- 選点（collocation points）の選び方
- 目的関数の各項の重み
- アーキテクチャ
  - ドメイン分割
  - ネットワーク構造 (MLP, CNN, Transformer, KAN, …)
  - 活性化関数
- カリキュラム学習
- 不確かさ定量化

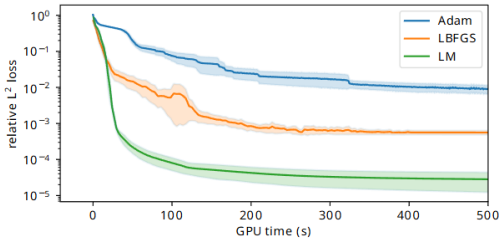


figure from [Bonfanti+ 2024]

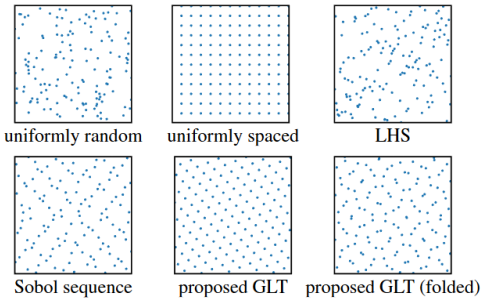


figure from [Matsubara & Yaguchi 2025]

see, e.g., [Wang+ arXiv:2308.08468]

Bonfanti+, The Challenges of the Nonlinear Regime for Physics-Informed Neural Networks, NeurIPS 2024

Matsubara & Yaguchi, Number Theoretic Accelerated Learning of Physics-Informed Neural Networks, AAAI 2025

Wang+, An expert's guide to training physics-informed neural networks, arXiv:2308.08468

Diffusion models / flow matching

- データ分布 ↔ ノイズ分布間の逐次的な変換を学習するタイプのモデル
- 柔軟な画像・軌道等の生成で強み

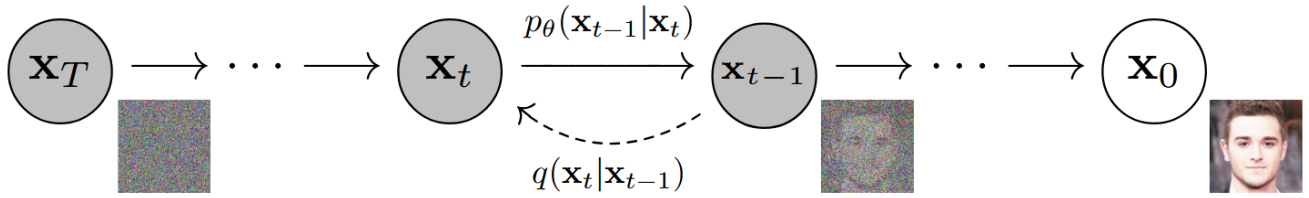
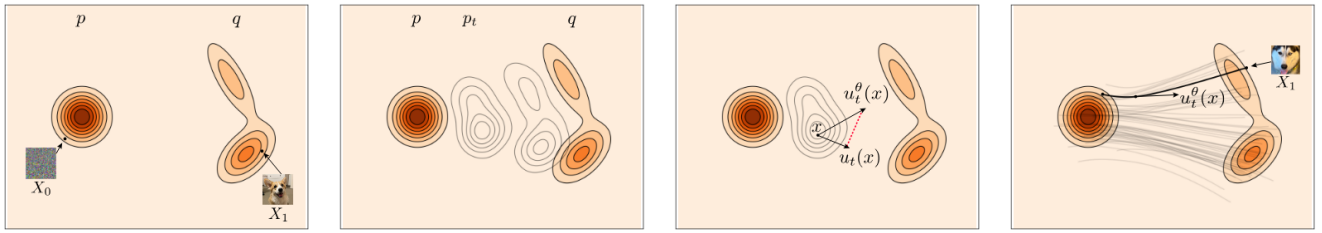


figure from [Ho+ 2020]



(a) Data. **(b)** Path design. **(c)** Training. **(d)** Sampling.

figure from Lipman+, Flow matching guide and code, arXiv:2412.06264

Ho+, Denoising Diffusion Probabilistic Models, NeurIPS 2020

Lipman+, Flow Matching for Generative Modeling, ICLR 2023

物理的な制約を満たすモデル学習

- シミュレータ出力に対して拡散モデルを学習しても、新しい生成結果が数理モデル（微分方程式など）を満たすとは限らない
- 最終的な生成結果が制約 $R(\boldsymbol{x}) = 0$ を満たすよう正則化しながら学習

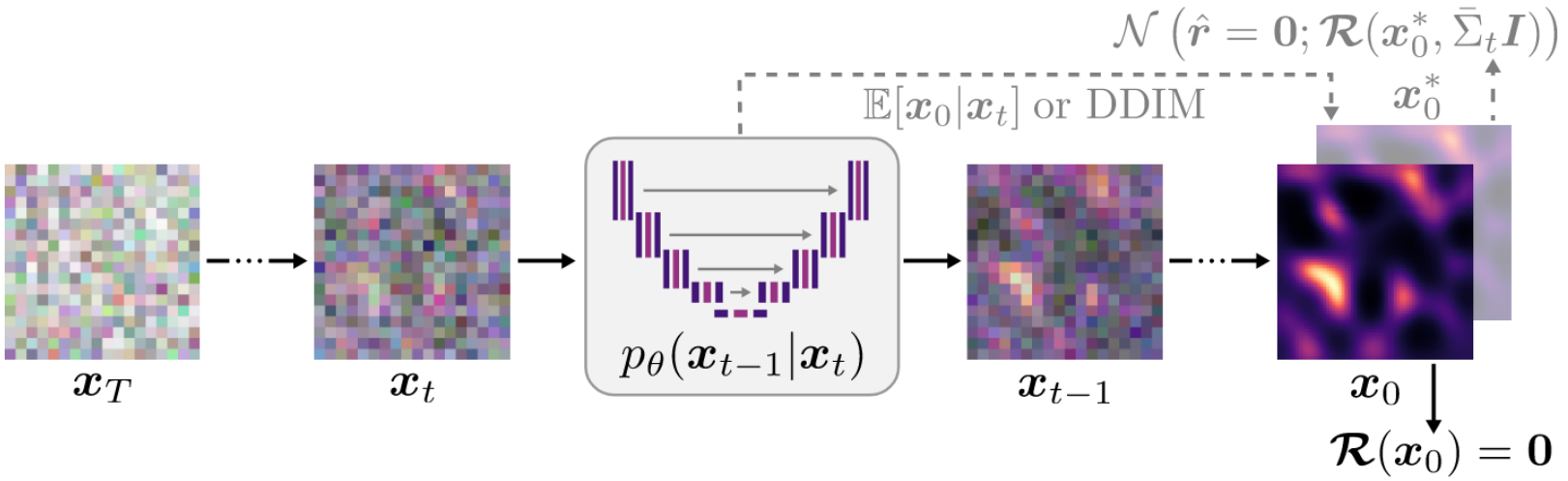


figure from [Bastek+ 2025]

物理的な制約を満たす生成プロセス

- まず、通常通り flow matching モデルを学習
- 物理的制約を満たすようサンプル生成; 各ステップ $t = \tau$ で、
  - まず最終時刻 $t = 1$ まで生成する
  - 生成結果を制約を満たす集合 $R(x) = 0$ に（なるべく）射影
    - Gauss-Newton法を1ステップだけ回す
  - フローを逆向きにたどって $t = \tau + \Delta t$ に戻る

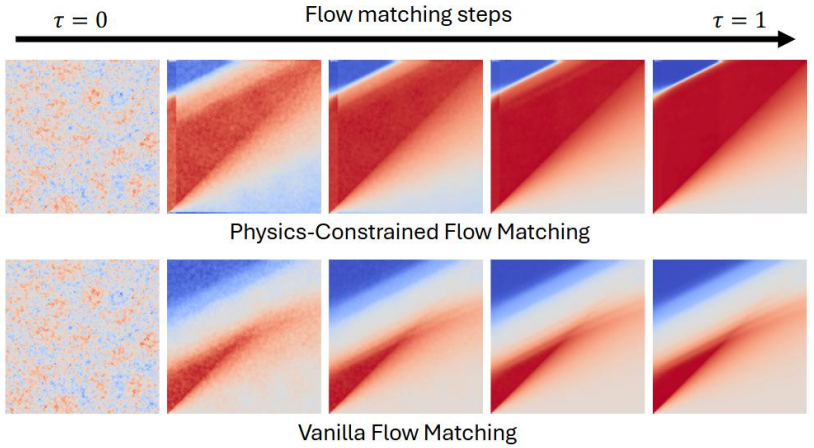


figure from [Utkarsh+ 2025]

Utkarsh+, Physics-Constrained Flow Matching: Sampling Generative Models with Hard Constraints, NeurIPS 2025

物理情報に基づく訓練データ設計

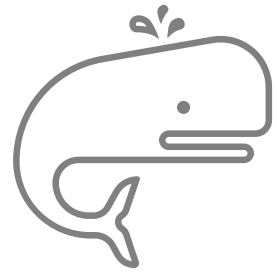
物理情報に基づく訓練データ

- シミュレータ $y = f(x)$ を使って訓練データ

$$\{(x_1, y_1 = f(x_1)), \ldots, (x_n, y_n = f(x_n))\}$$

  を生成

- 訓練データを介して、シミュレータからの情報が間接的に学習される
- 例
  - 解作用素の学習（neural operators）
  - 機械学習による逆問題解法（simulation-based inference）



(解)作用素の学習

- 例えば「初期条件 → 時刻 t での解」の作用素を学習
  - 他の関数ペアでもよい [e.g., Li+ 2021; Kovachki+ 2023; Lu+ 2021]
  - データは数値計算で作る
  - 関数から関数への写像をニューラルネットで表現

$u(0, \boldsymbol{x}) = a(\boldsymbol{x})$ $u(t, \boldsymbol{x})$

Huboedeker, wikimedia commons

$$\frac{\partial u(t, \boldsymbol{x})}{\partial t} - Lu(t, \boldsymbol{x}) = 0 \quad \text{for} \quad t \in [0, T], \ \boldsymbol{x} \in \Omega$$

$$u(0, \boldsymbol{x}) = a(\boldsymbol{x}) \quad \text{for} \quad \boldsymbol{x} \in \Omega$$

Li+, Fourier Neural Operator for Parametric Partial Differential Equations, ICLR 2021

Kovachki+, Neural Operator: Learning Maps Between Function Spaces With Applications to PDEs, *J. Machine Learning Research*, 2023

Lu+, Learning Nonlinear Operators via DeepONet Based on the Universal Approximation Theorem of Operators, *Nature Machine Intelligence*, 2021

Fourier neural operators [Li+ 2021]

- 入力関数から出力関数へのmapを複数のkernel integral operatorsで表す

$$v_{i+1}(y) = \sigma\left(\int K^{(i)}(x,y)v_i(x)dx + b^{(i)}(y)\right)$$

- translation-invariant kernel $K^{(i)}(x,y) = k^{(i)}(x,y)$ を使って、積分をFFT+畳み込みで高速に計算できるようにする

$$v_{i+1} = \sigma\left(W_i v_i + \mathcal{F}^{-1}\left(R_i \cdot \mathcal{F}(v_i)\right) + b_i\right)$$

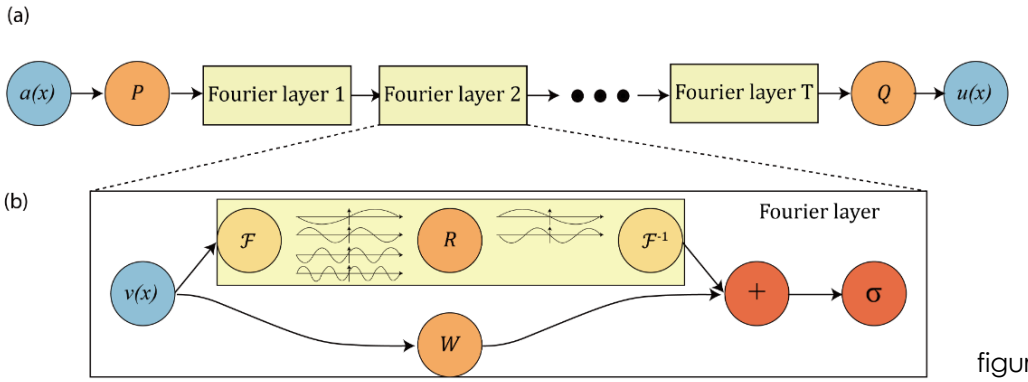


figure from [Li+ 2021]

Li+, Fourier Neural Operator for Parametric Partial Differential Equations, ICLR 2021

DeepONet [Lu+ 2021]

- Branch net: 入力関数の評価点 $x_1, \ldots, x_m$ での値をencode
  - $b_1\big(u(x_1), \ldots, u(x_m)\big),\ \ldots,\ b_p\big(u(x_1), \ldots, u(x_m)\big)$ の p 次元の潜在表現ができる
- Trunk net: ↑潜在表現を出力関数の評価点 y についてdecode

Theorem 1 (Universal Approximation Theorem for Operator). *Suppose that σ is a continuous non-polynomial function, X is a Banach Space, $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$ are two compact sets in X and $\mathbb{R}^d$, respectively, V is a compact set in $C(K_1)$, G is a nonlinear continuous operator, which maps V into $C(K_2)$. Then for any $\epsilon > 0$, there are positive integers n, p, m, constants $c_i^k, \xi_{ij}^k, \theta_i^k, \zeta_k \in \mathbb{R}$, $w_k \in \mathbb{R}^d$, $x_j \in K_1$, $i = 1, \ldots, n$, $k = 1, \ldots, p$, $j = 1, \ldots, m$, such that*

$$\left| G(u)(y) - \sum_{k=1}^{p} \underbrace{\sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k \right)}_{branch} \underbrace{\sigma(w_k \cdot y + \zeta_k)}_{trunk} \right| < \epsilon \qquad (1)$$

holds for all $u \in V$ and $y \in K_2$.

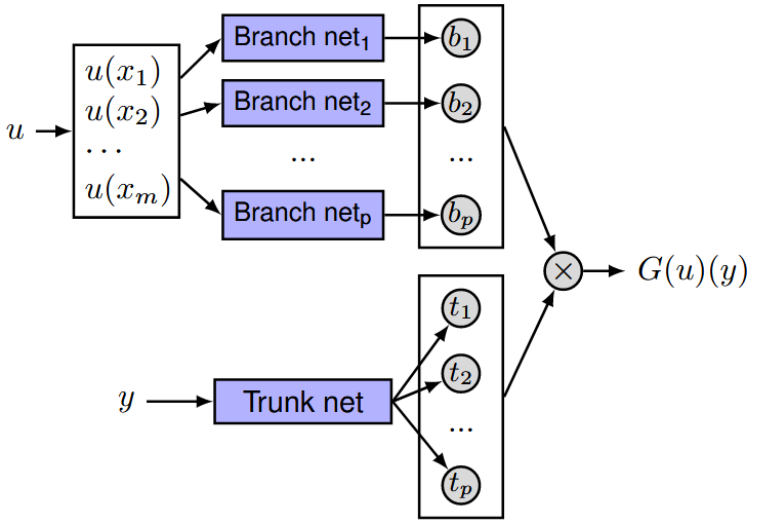
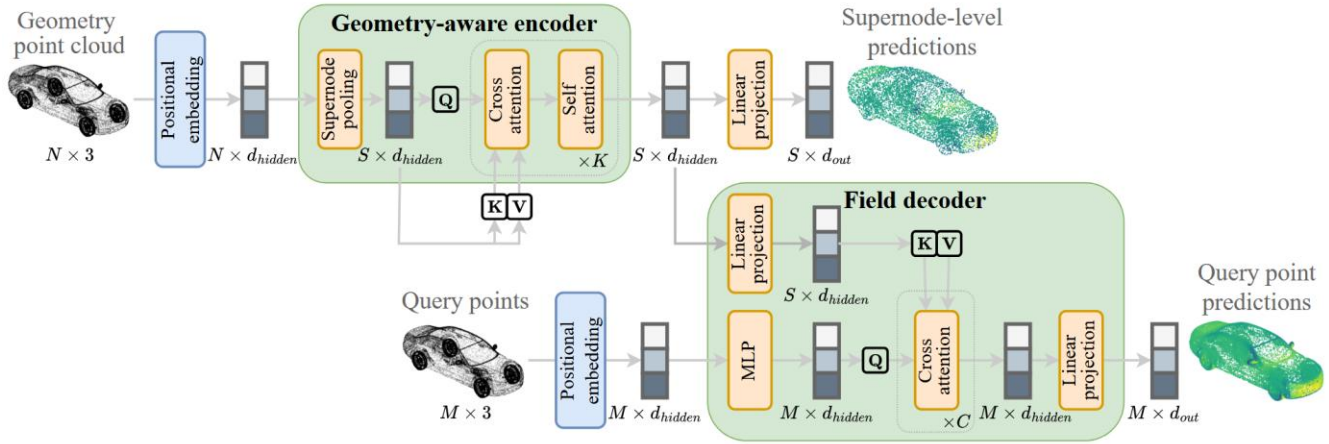


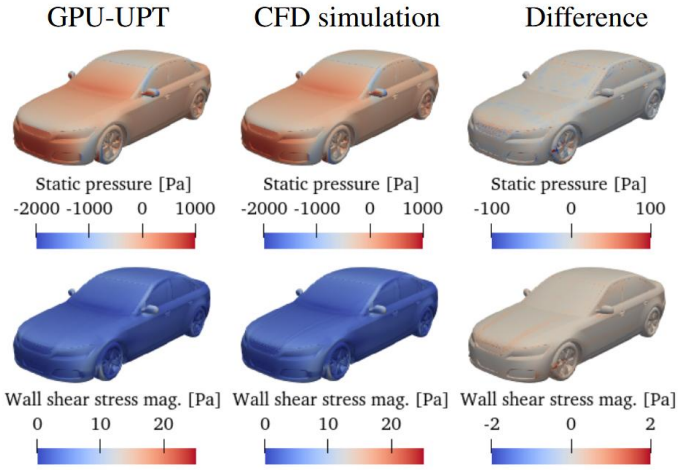
figure from [Lu+ 2021]

Lu+, Learning Nonlinear Operators via DeepONet Based on the Universal Approximation Theorem of Operators, *Nature Machine Intelligence*, 2021

最近の展開

- 複雑な形状まわりの流れ場を予測 [Bleeker+ 2025]
  - 形状を点群で表現する
  - Transfomerベースのアーキテクチャ



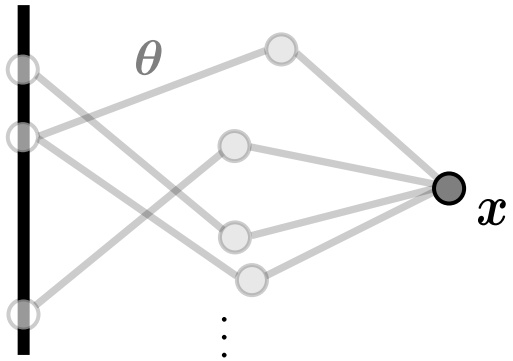


figures from [Bleeker+ 2025]

Bleeker+, NeuralCFD: Deep learning on high-fidelity automotive aerodynamics simulations, arXiv:2502.09692

逆問題、特に劣決定な逆問題

- ある観測 x を実現するパラメタ θ がいくつも（無数に）ある

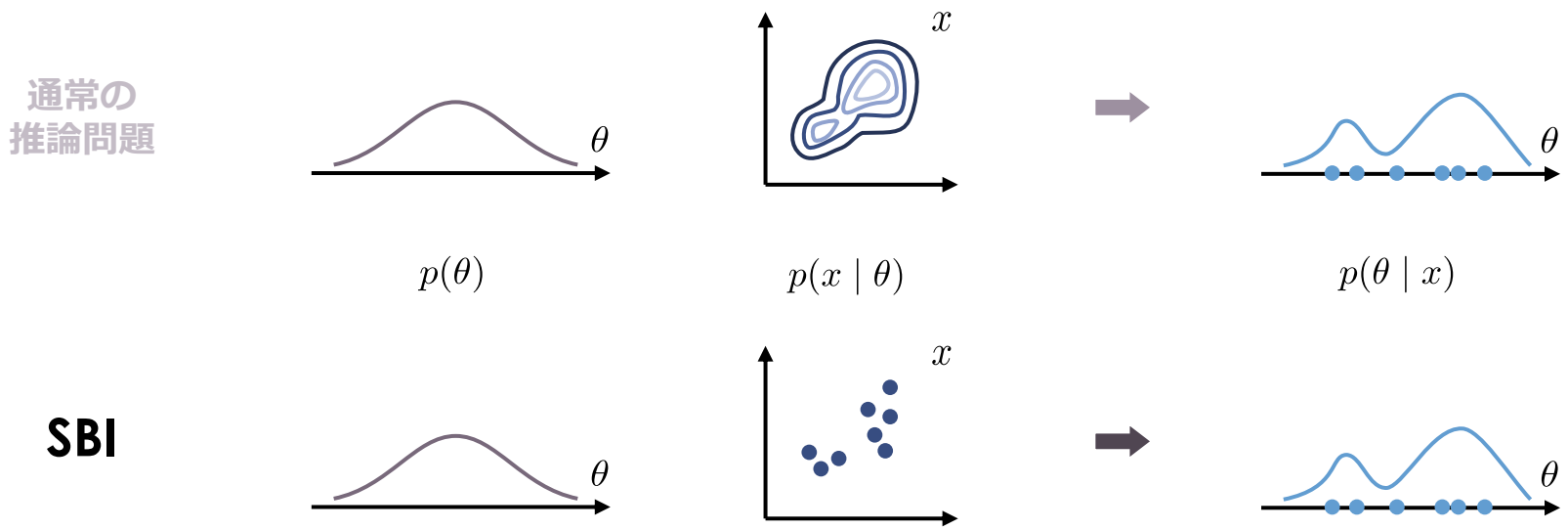


- 正則化　　何らかの規準 $g(\boldsymbol{\theta})$ を設定して、「好ましい」$\boldsymbol{\theta}$ を選択
- **ベイズ推論**　事前分布 $p(\boldsymbol{\theta})$ を設定して、事後分布 $p(\boldsymbol{\theta} \mid \boldsymbol{x})$ を推論

$$p(\boldsymbol{\theta} \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{x})}$$

Simulation-based inference

- 尤度関数 $p(\boldsymbol{x} \mid \boldsymbol{\theta})$ の値を計算せず、事後分布 $p(\boldsymbol{\theta} \mid \boldsymbol{x})$ を推論する問題
  - 「尤度なし推論（likelihood-free inference）」 または
    「シミュレーションに基づく推論（simulation-based inference; **SBI**）」



Cranmer+, The frontier of simulation-based inference, *Proc. the National Academy of Sciences*, 2020

深層学習によるSBI: 事後分布を直接学習（NPE）

- 観測 x_o が与えられたとき、事後分布 $p(\theta \mid x = x_o)$ （からのサンプル）を得たい
- Neural posterior estimation (NPE)

基本的には
一度だけ実行

Input 事前分布，シミュレータ
Output ニューラルネットによる事後分布の近似 $q_\phi(\theta \mid x)$
1 $i = 1, \ldots, N$ について、
2 事前分布とシミュレータから、 (θ_i, x_i) を生成して保存
3 保存した $\{(\theta, x)\}$ から、事後分布 $q_\phi(\theta \mid x)$ （normalizing flowsなど）を学習

↓

Input 観測 x_o，近似した事後分布 $q_\phi(\theta \mid x)$
Output 事後分布の評価値 $q(\theta \mid x = x_o)$
1 $q(\theta \mid x = x_o)$ を計算

深層学習によるSBIの利点

- 深層生成モデルで複雑な事後分布を表現できる
  - normalizing flow, diffusion model, flow matching, …
- シミュレーションの実行は、基本的にはデータ生成時のみに起こる
  - 新たな観測に対する事後分布推論は、NNの実行だけ
  - 新たな観測に対してデータ生成とモデル訓練を都度行う方法もある (sequential SBI)
- ちなみに: PINNも逆問題に使える
  - 実際、多くの研究でPINNによる「逆問題」が扱われている
  - neural likelihood estimationの特殊な場合といえる
  - PINNに未知パラメタも入力して、同時に推定／後から最適化

SBIの応用例

- 物体把持のためのロボット姿勢推論 [Marlier+ arXiv:2109.14275]
  - デプス画像を入力して、物体把持するためのアームの位置姿勢（の分布）を出力
  - シミュレーションでランダムにアームを動かして成功した例をデータとして保存
  - domain randomziation

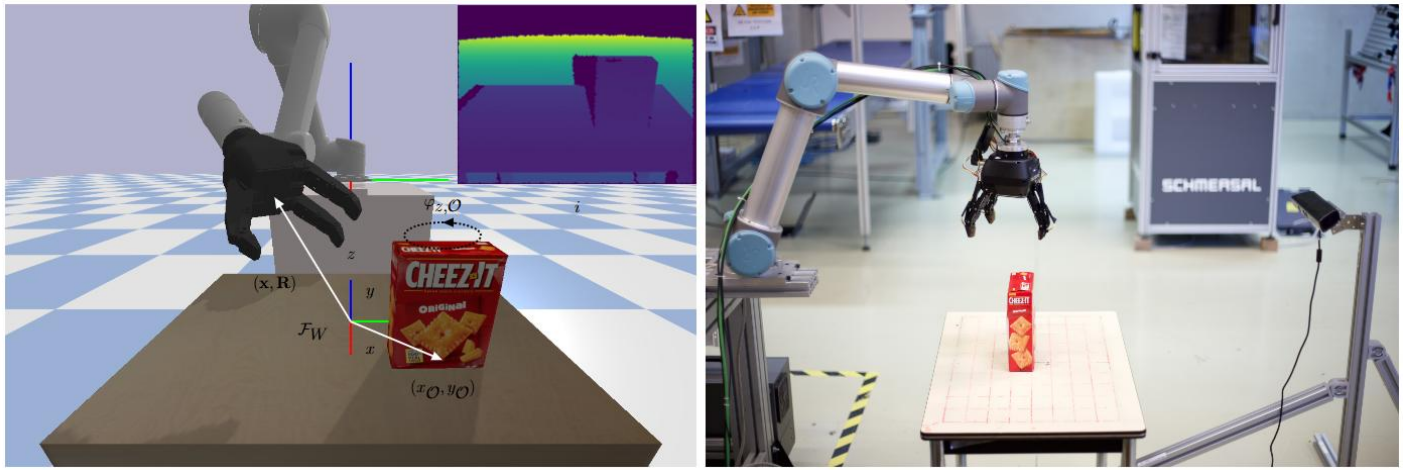


figure from [Marlier+ arXiv:2109.14275]

Marlier+, Simulation-based Bayesian inference for multi-fingered robotic grasping, arXiv:2109.14275

SBIの応用例

- diffusion modelベースの状態量推定による全球の気象再解析

[Andry+, arXiv:2504.18720]

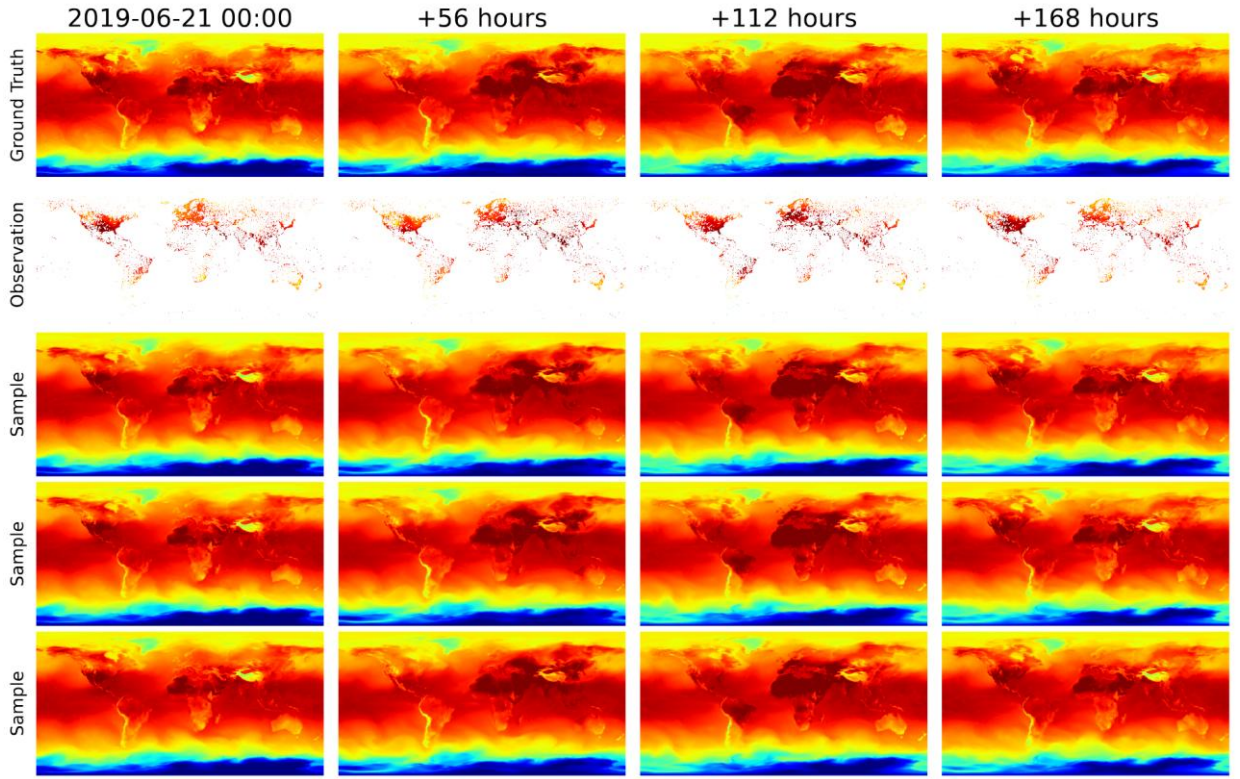


figure from [Andry+ arXiv:2504.18720]

Andry+, Appa: Bending weather dynamics with latent diffusion models for global data assimilation, arXiv:2504.18720

物理情報に基づくモデル

物理情報に基づくモデル

- 機械学習モデル

（ニューラルネット, 決定木, カーネル法, …）

  - 大量のパラメタをもつことがある
  - 同質かつ多量のデータが必要
  - 実際のデータに適応できる
  - 解釈しにくい

- 科学モデル

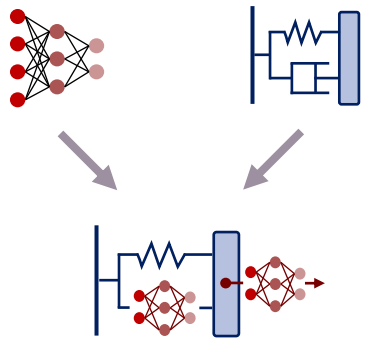
（微分方程式, 数値計算, グラフ, 論理式, …）

  - 少数のパラメタで表現される（べき）
  - 多様な少量のデータの蓄積
  - 抽象化・理想化のために
    実際のデータと合わないことがある
  - 理解に貢献

- 両者の「良いとこ取り」ができないか？
  - 予測をよりよく
  - 解釈しやすく

  } できる**かも**

  - 融合の程度は様々
    - 「物理で出てくる構造っぽいのを使う」から
      「**具体的な微分方程式を使う**」まで



Hamiltonian neural networks [Greydanus+ 2019]

- ハミルトニアンをニューラルネットで構成

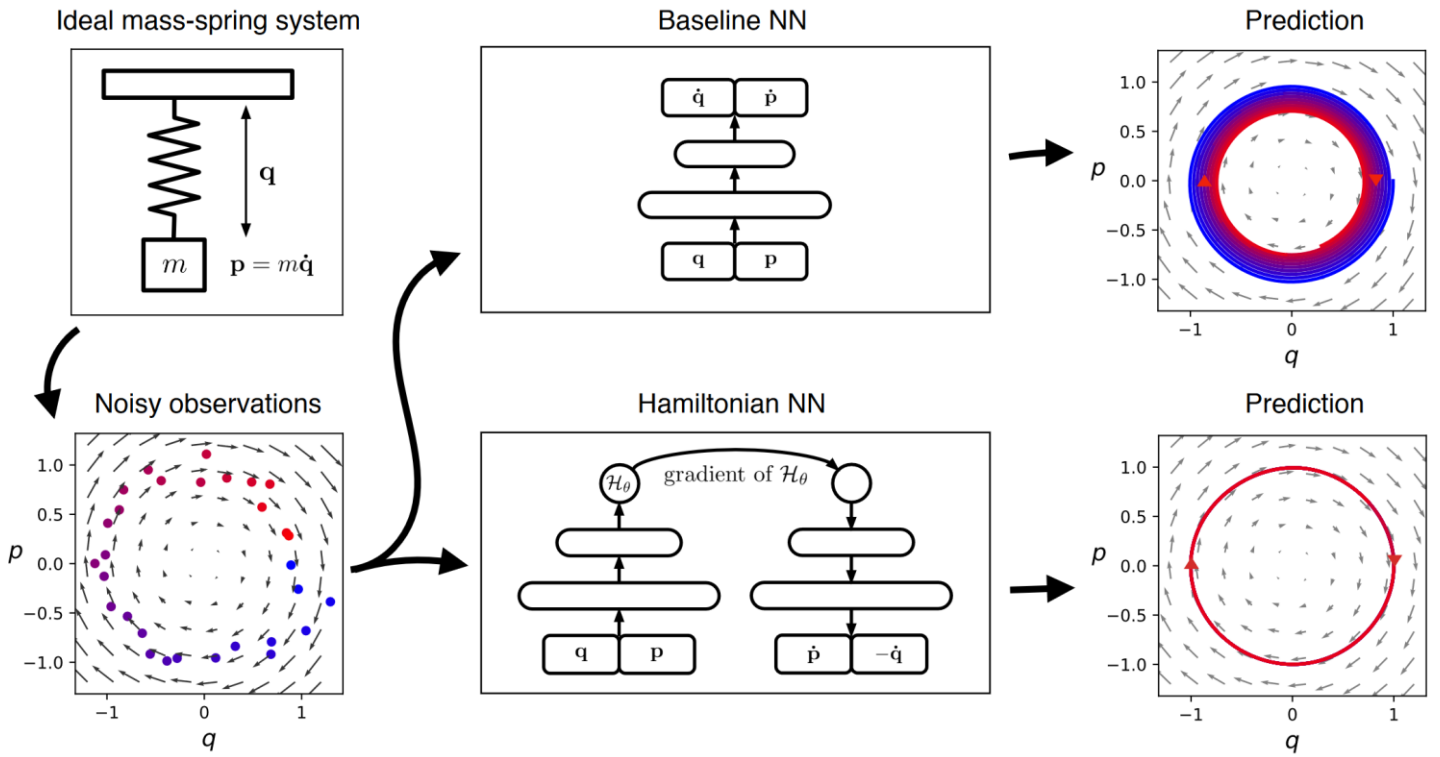


figure from [Greydanus+ 2019]

COVID-19 感染者数予測 [Arık+ 2020]

- SIRモデルの係数を移動者数などの統計データから予測

$$
\begin{aligned}
S[t] &= S[t-1] - (\beta^{(d)} \cdot I^{(d)}[t-1] + \beta^{(u)} \cdot I^{(u)}[t-1]) \cdot S[t-1]/N[t-1] + \eta \cdot (R^{(d)}[t-1] + R^{(u)}[t-1]), \\
E[t] &= E[t-1] + (\beta^{(d)} \cdot I^{(d)}[t-1] + \beta^{(u)} \cdot I^{(u)}[t-1]) \cdot S[t-1]/N[t-1] - \alpha \cdot E[t-1], \\
I^{(u)}[t] &= I^{(u)}[t-1] + \alpha \cdot E[t-1] - (\rho^{(I,u)} + \gamma) \cdot I^{(u)}[t-1], \\
I^{(d)}[t] &= I^{(d)}[t-1] + \gamma \cdot I^{(u)}[t-1] - (\rho^{(I,d)} + \kappa^{(I,d)} + h) \cdot I^{(d)}[t-1], \\
R^{(u)}[t] &= R^{(u)}[t-1] + \rho^{(I,u)} \cdot I^{(u)}[t-1] - \eta \cdot R^{(u)}[t-1], \\
R^{(d)}[t] &= R^{(d)}[t-1] + \rho^{(I,d)} \cdot I^{(d)}[t-1] + \rho^{(H)} \cdot (H[t-1] - C[t-1]) - \eta \cdot R^{(d)}[t-1], \\
H[t] &= H[t-1] + h \cdot I^{(d)}[t-1] - (\kappa^{(H)} + \rho^{(H)}) \cdot (H[t-1] - C[t-1]) - \kappa^{(C)} \cdot (C[t-1] - V[t-1]) - \kappa^{(V)} \cdot V[t-1], \\
C[t] &= C[t-1] + c \cdot (H[t-1] - C[t-1]) - (\kappa^{(C)} + \rho^{(C)} + v) \cdot (C[t-1] - V[t-1]) - \kappa^{(V)} \cdot V[t-1], \\
V[t] &= V[t-1] + v \cdot (C[t-1] - V[t-1]) - (\kappa^{(V)} + \rho^{(V)}) \cdot V[t-1], \\
D[t] &= D[t-1] + \kappa^{(V)} \cdot V[t-1] + \kappa^{(C)} \cdot (C[t-1] - V[t-1]) + \kappa^{(H)} \cdot (H[t-1] - C[t-1]) + \kappa^{(I,d)} \cdot I^{(d)}[t-1],
\end{aligned}
$$

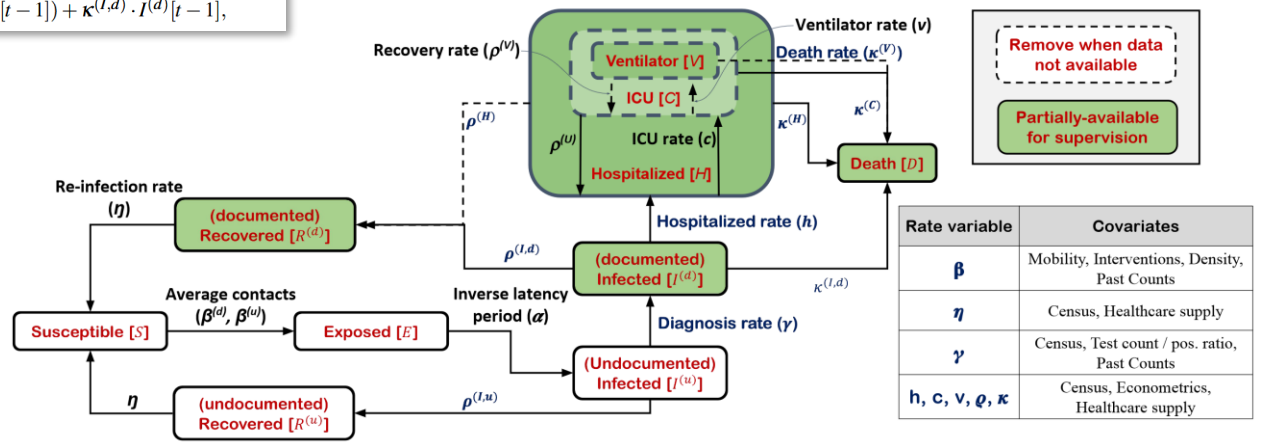
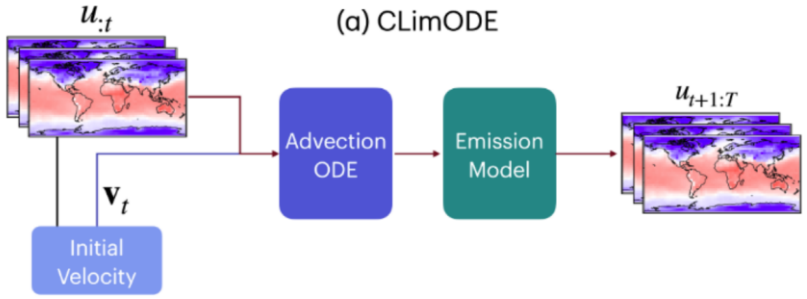


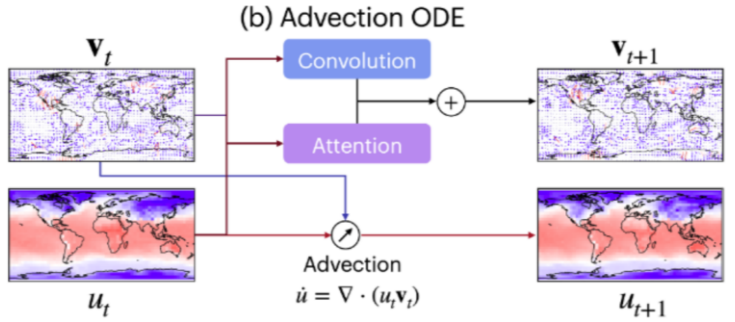
figure from [Arık+ 2020]

Arık+, Interpretable Sequence Learning for COVID-19 Forecasting, NeurIPS 2020

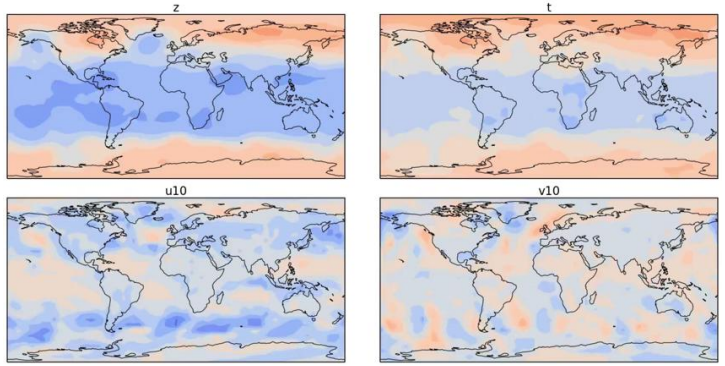
気象予測 [Verma+ 2024]

- 予測値が移流方程式に従うNN





figures from [Verma+ 2024]



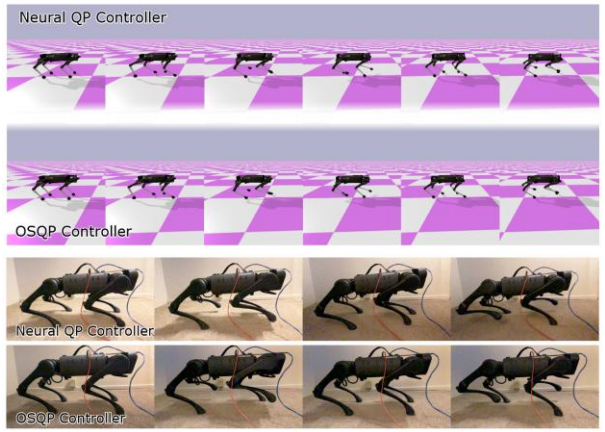
video from the project page of [Verma+ 2024]
https://yogeshverma1998.github.io/ClimODE/

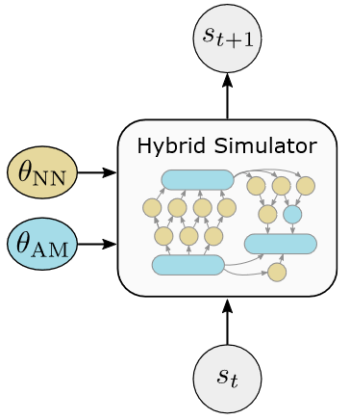
Verma+ ClimODE: Climate and Weather Forecasting with Physics-Informed Neural ODEs, ICLR 2024

接触などのある剛体シミュレーション

- ロボットなどのシミュレーションでは、接触判定、摩擦力、空気力などの精緻な再現が難しい
  - 難しい部分は機械学習でデータから獲得

[Ajay+ 2018; Hwangbo+ 2019; Zeng+ 2019; Golemo+ 2018; Heiden+ 2021]





figures from [Heiden+ 2021]

Ajay+, Augmenting Physical Simulators with Stochastic Neural Networks: Case Study of Planar Pushing and Bouncing, IROS 2018
Hwangbo+, Learning Agile and Dynamic Motor Skills for Legged Robots, Science Robotics, 2019
Zeng+, TossingBot: Learning to Throw Arbitrary Objects with Residual Physics, RSS 2019
Golemo+, Sim-to-Real Transfer with Neural-Augmented Robot Simulation, CoRL 2018
Heiden+, NeuralSim: Augmenting Differentiable Simulators with Neural Networks, ICRA 2021

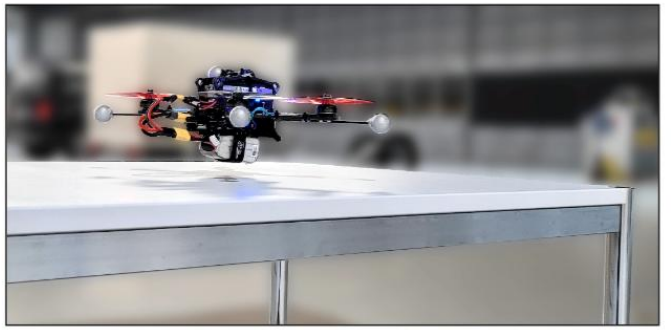
クワッドロータのモデリング

- 剛体運動の運動方程式 + 空気力を予測する機械学習モデル

$$f(\boldsymbol{x}, \boldsymbol{u}) = f_{\mathcal{F}}(\boldsymbol{x}, \boldsymbol{u}) + f_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{u})$$

$$f_{\mathcal{F}}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{v}_W \\ \boldsymbol{q}_{WB} \cdot \begin{bmatrix} 0 \\ \boldsymbol{\omega}_B/2 \end{bmatrix} \\ \frac{1}{m} \boldsymbol{q}_{WB} \odot \boldsymbol{T}_B + \boldsymbol{g}_W \\ \boldsymbol{J}^{-1} (\boldsymbol{\tau}_B - \boldsymbol{\omega}_B \times \boldsymbol{J}\boldsymbol{\omega}_B) \end{bmatrix}$$

$$f_{\mathcal{D}}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{0}_2 \\ f_{\mathcal{D}_\theta}(\boldsymbol{x}, \boldsymbol{u}) \\ f_{\mathcal{D}_\psi}(\boldsymbol{x}, \boldsymbol{u}) \end{bmatrix}$$



(a)

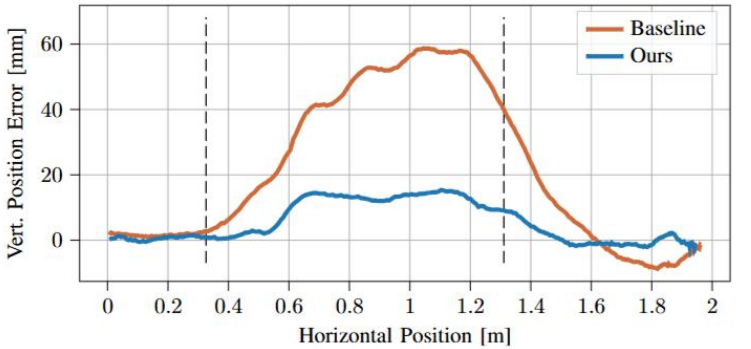


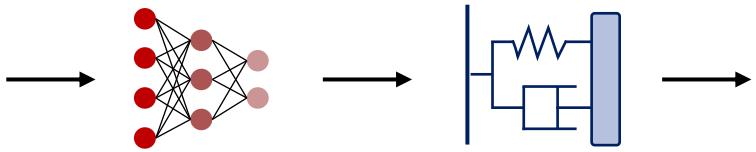
figure from [Saltzmann+ 2023]

Torrente+, Data-Driven MPC for Quadrotors, *IEEE Robotics and Automation Letters*, 2021

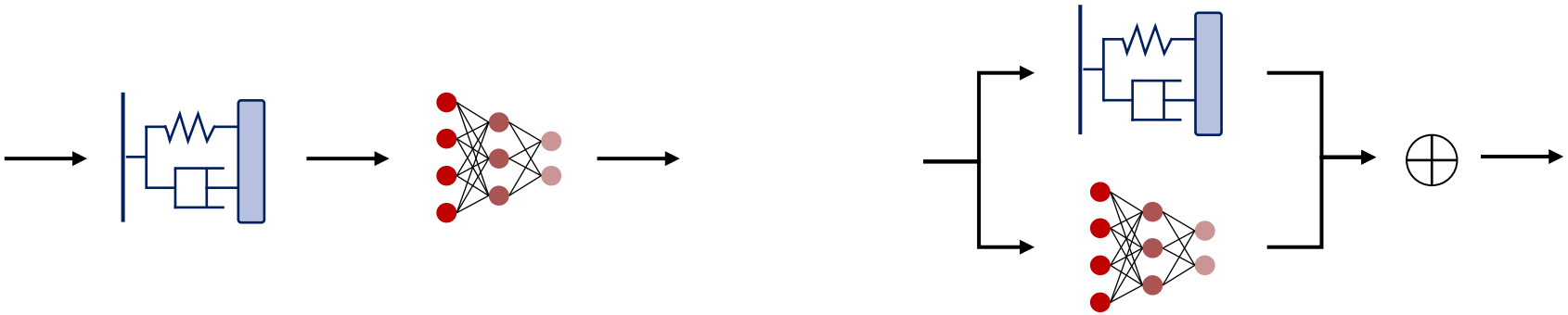
Saltzmann+, Real-time Neural-MPC: Deep Learning Model Predictive Control for Quadrotors and Agile Robotic Platforms, *IEEE Robotics and Automation Letters*, 2023

ハイブリッドモデルのつくり方

- 科学モデルが「後 / 外側」
  - 機械学習による feature extraction / parametrization / submodel / …

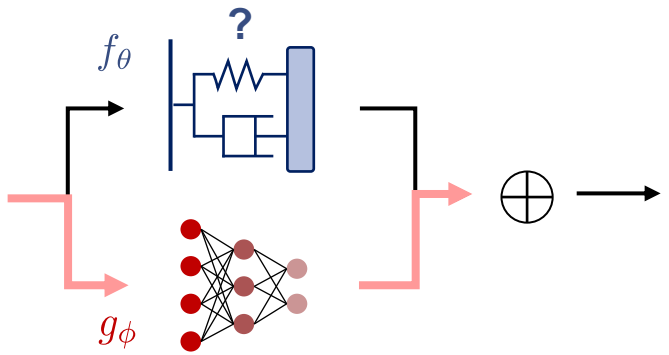


- 科学モデルが「先 / 内側」
  - 機械学習による closure / residual physics / discrepancy modeling [Kaheman+ 2019]
    Δ-ML [Ramakrishnan+ 2015] / …



科学モデルの未知パラメタ推定

- 機械学習モデルが「外」にあるとき、科学モデルの未知パラメタがうまく推定できない可能性
  - 機械学習モデルだけでデータにフィットできてしまう

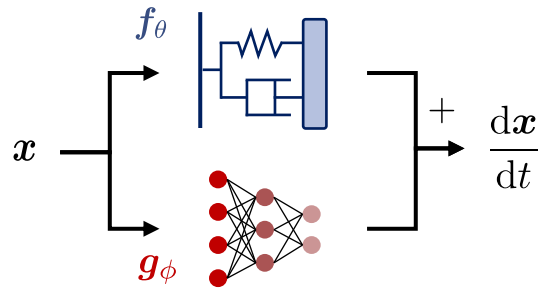


- 例えば、 $y = f_\theta(x) + g_\phi(x)$
  - $g_\phi(x)$ が $y - f_\theta(x)$ にフィットして、θ はなんでも良い、となってしまう
  - $g_\phi(x)$ の**過剰な表現力を抑える**べき

制約 / 正則化によるハイブリッドモデル学習 1/2

- 加法的ハイブリッドモデルの学習 [Yin+ 2021]
  - 特に、ハイブリッドな neural ODE:

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = \boldsymbol{f}_\theta(\boldsymbol{x}) + \boldsymbol{g}_\phi(\boldsymbol{x}) \quad \ldots \ (\star)$$



  - 機械学習モデル $\boldsymbol{g}_\phi$ のノルムがなるべく小さくなるように学習

訓練データ

$$\underset{\theta,\phi}{\text{minimize}} \ \|\boldsymbol{g}_\phi\| \ \text{ s.t. } \ \forall \boldsymbol{X} \in \mathcal{D}, \ \boldsymbol{X} = \text{ODESolve}(\star)$$

訓練データ中のエピソード

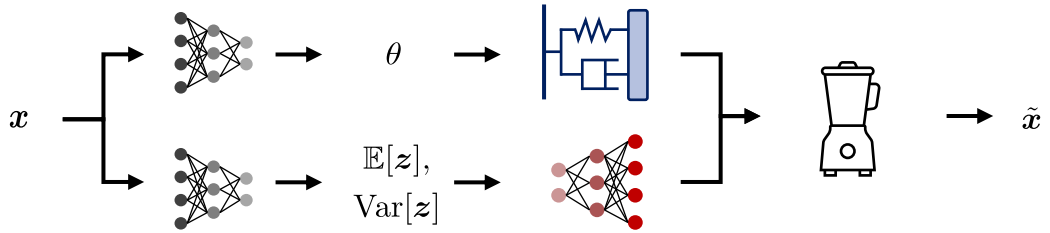
初期値問題ソルバ
（初期値は所与としたり、推定したり）

- 拡張ラグランジュ法で解く
- 関数ノルムで「表現力」をあらわすという考え方； 自然だが、本当にそれだけか？

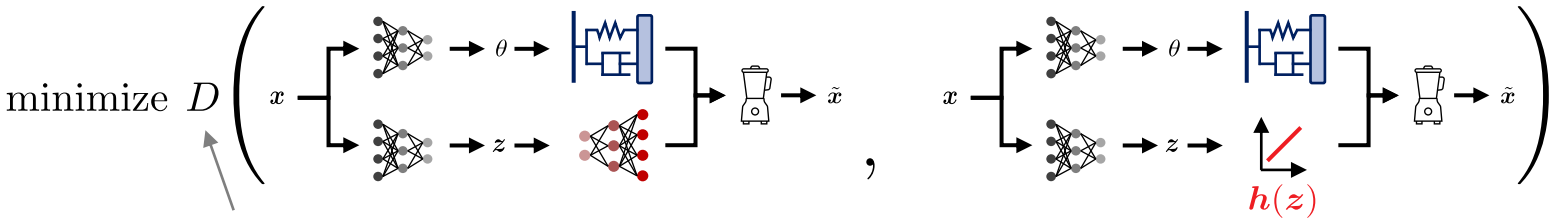
Yin+, Augmenting Physical Models with Deep Networks for Complex Dynamics Forecasting, ICLR 2021

制約 / 正則化によるハイブリッドモデル学習 2/2

- より一般の結合方法を想定 [Takeishi & Kalousis 2021]
  - 特に、デコーダ部分がハイブリッドな(変分)オートエンコーダ



  - ハイブリッドモデルと「科学モデルだけの場合」の差を小さくなるよう正則化



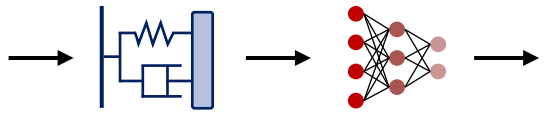
分布間の非類似度

NNを簡素なモデルで置き換える
（例えば $h(z) = 0, h(z) = Az + b$）

sharpness-aware minimizationによる学習

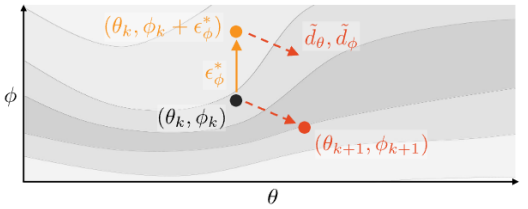
- 一般のハイブリッドモデル

$$y = g_\phi\left(f_\theta(x); x\right)$$



の学習

- 損失関数が ϕ についてなるべく平坦な局所解を探す [Takeishi arXiv:2602.06837]
  - 損失が平坦 = パラメタの表現に必要な情報量が少ない
    = その意味でモデルがシンプル [Hochreiter & Schmidhuber 1997]
  - sharpness-aware minimization [Foret+ 2021] を使う



- 普通に学習するより θ をあてやすい
- モデル構造によらず適用可能

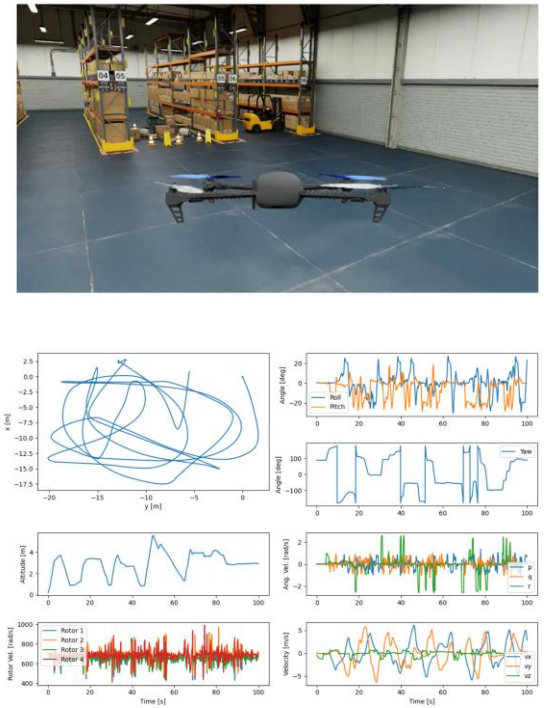
Takeishi, Learning Deep Hybrid Models with Sharpness-Aware Minimization, arXiv:2602.06837

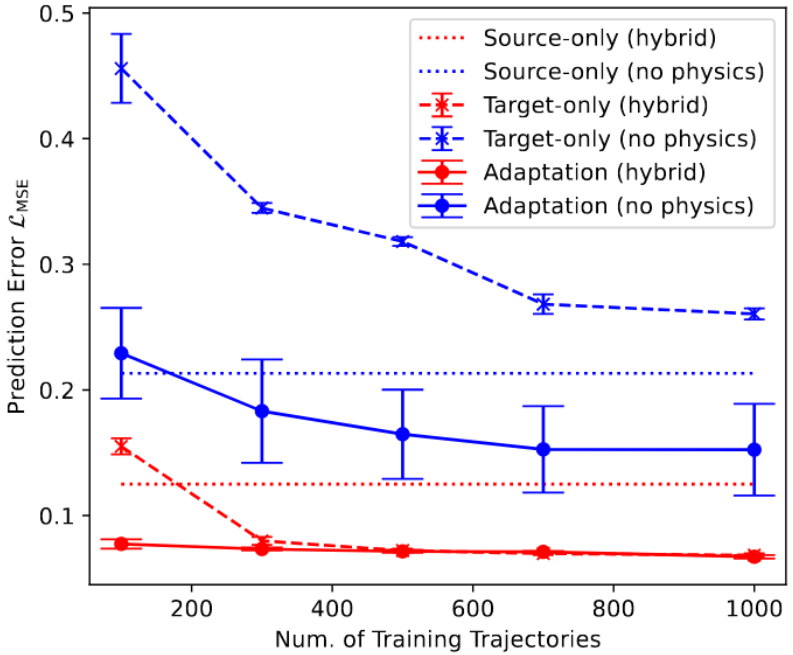
Hochreiter & Schmidhuber, Flat Minima, Neural Computation, 1997

Foret+, Sharpness-Aware Minimization for Efficiently Improving Generalization, ICLR 2021

クワッドロータのモデリング 再び

- 通常のNNよりハイブリッドモデルのほうが転移学習の設定でも有利





Osaka+, Deterministic and Stochastic Hybrid Modeling with Regularization, SMC 2025

Osaka+, Domain Adaptation with Hybrid Modeling for Learning Dynamical Systems, ACML 2025

まとめ

- **物理学的な事前知識**を活用しつつ**機械学習**でデータに基づいてモデリングを行う方法について、方法論の立場から概観する
  - 物理知識に基づく **① 訓練データ、② モデル、③ 目的関数** の設計

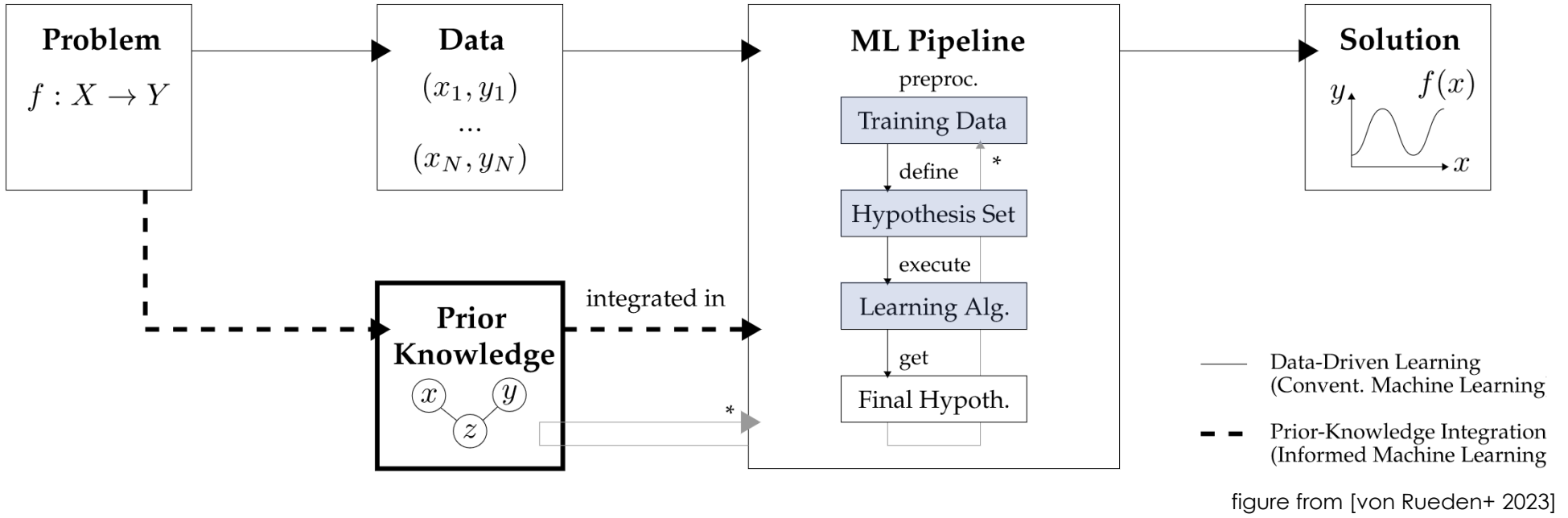


figure from [von Rueden+ 2023]

von Rueden+, Informed Machine Learning – A Taxonomy and Survey of Integrating Knowledge into Learning Systems, *IEEE Transactions on Knowledge and Data Engineering*, 2023