J. Plasma Fusion Res. Vol.102, No.4 (2026) 137-144

講座 ニューラルネットワークを用いた物理モデリング・シミュレーション

4. 物理モデリングのためのニューラルネットワーク

4. Neural Networks for Physical Modelling

谷口 隆晴
YAGUCHI Takaharu
神戸大学理学研究科数学専攻

（原稿受付：2025 年 7 月 18 日）

近年，深層学習を利用して，物理現象に関する観測データから，その現象を記述するための微分方程式を導出する手法が注目されている．このような方法は，現象をモデル化するだけでなく，データとしてシミュレーターから得られた数値計算結果を利用すれば，シミュレーターに対するサロゲートモデルの構築手法ともなり得る．本章では，このような方法のうち，代表的なものについて説明する．

Keywords:
Hamiltonian neural network, Lagrangian neural network, neural symplectic form, neural ordinary differential equation, Kolmogorov-Arnold network

2.1 はじめに

近年の深層学習の発展は目覚ましく，その応用は，物理学の様々な問題にも及んでいる．本章では，特に，物理モデリングのためのニューラルネットワークの応用について説明する．具体的な問題設定としては，未知の微分方程式に従うと考えられる物理現象に対して，その状態変数および時間微分についてのデータが得られているときに，その現象を表す微分方程式を推定するという問題を考える．

このような問題は，特にシステム解析の分野でシステム同定として考えられていた問題と同じものである．システム同定では，モデル化対象のシステムについての観測データが得られているときに，そのシステムを表す微分方程式や伝達関数などを推定する．古典的なシステム同定手法は，主に線形の方程式を対象としており，いわゆる状態空間モデルなどの形で支配方程式を推定する手法が研究されてきた．非線形なシステムに対しても，同様に，研究は進められており，ニューラルネットワークを利用した方法も知られていた．しかし，高性能な GPU などの計算機に支えられた現在のニューラルネットワークとは異なり，計算量などの観点から，複雑なモデルは利用しにくかったものと思われる．

これに対して，2018 年にニューラル常微分方程式 [1] が提案され，また，その翌年にハミルトニアンニューラルネットワーク [2] と呼ばれる，データからハミルトン方程式の形で運動方程式を推定する方法が提案されると，その後，数年間で様々な拡張が提案された．現在では，単純にデータから方程式を推定するだけでなく，保存則や対称性を発見する方法や，散逸系をモデル化するための方法，偏微分方程式をモデル化する方法などが提案されている．

2.2 ニューラル常微分方程式

まず，本章では，ニューラルネットワークとして，単純な多層パーセプトロンを使う方法を考える．多層パーセプトロンは，アフィン変換と，活性化関数と呼ばれる非線形関数を繰り返し合成した関数であり，最も基本的なニューラルネットワークである．例えば，行列 A_1, A_2，ベクトル b_1, b_2，活性化関数 $\sigma\colon R \to R$ を利用して定義される

$$f_{NN}\colon R^N \to R^M, \quad f_{NN}(x) = A_2\sigma\left(A_1 x + b_1\right) + b_2 \quad (1)$$

という関数は多層パーセプトロンの例である．ただし，活性化関数 σ にベクトルが入力された場合には，ベクトルの各成分に σ を適用することとする．また，行列やベクトルのサイズは，計算ができるようなものであれば，どのようなものでも構わない．

多層パーセプトロンは，普遍近似性と呼ばれる性質をもつことが知られており，少なくとも，理論的には多層パーセプトロンで十分に広いクラスの物理現象がモデル化できると言える．普遍近似性とは，十分に大きい行列やベクトルと，多項式でない活性化関数を利用すれば，有界閉集合上で定義された任意の連続関数が近似できるという性質である．また，連続関数だけでなく，より滑らかな関数についても，活性化関数を適切に選べば，微分も含めて近似できることも知られている．つまり，概ね，ほとんどの関数は，その微分も含めて多層パーセプトロンで近似できると考えてよい．したがって，物理現象を記述する方程式を表すための式も，同様に近似できると期待される．

Graduate School of Science and Faculty of Science, Kobe University, Kobe, HYOGO 657-8501, Japan
author's e-mail: yaguchi@pearl.kobe-u.ac.jp

この性質を踏まえると，最も単純なモデルとして，現象を表す状態変数 $u\colon t \in R \mapsto u(t) \in R^N$ が常微分方程式

$$\frac{\mathbf{d}\boldsymbol{u}}{\mathbf{d}\boldsymbol{t}} = \boldsymbol{f}(\boldsymbol{u}) \tag{2}$$

に従うと仮定して，右辺の f をニューラルネットワーク f_{NN} で推定する方法が思いつくであろう．このようなモデルをニューラル常微分方程式と呼ぶ [1].

より具体的には，様々な初期値から時間発展した，状態変数 u とその微分 $\frac{\mathrm{d}u}{\mathrm{d}t}$ の値が，様々な時刻において観測されており，データとして与えられていると仮定する．このデータを $\{(u_1, \dot{u}_1), (u_2, \dot{u}_2), \ldots, (u_m, \dot{u}_m)\}$ と表すことにしよう．m はデータ数であり，$\dot{u}_j$ は時間微分のデータを表す．ニューラル常微分方程式では，通常，データに対する予測誤差

$$\sum_{j=1}^{m} ||\dot{u}_j - f_{NN}(u_j)||^2 \tag{3}$$

が小さくなるように学習，すなわち，ニューラルネットワークを定めるためのパラメータである，行列やベクトルを調整する．前述の普遍近似性から，十分大きいニューラルネットワークを利用すれば，実際に誤差を小さくする関数 f_{NN} が存在することがわかっている．実際，このような方法で，f_{NN} をデータから求めると，非常に高い精度で方程式を近似することができる．

注意：ニューラル常微分方程式は，もともとは，ある種の構造をもつニューラルネットワークを常微分方程式とみなして，データ同化などで利用されている Adjoint 法を使って学習を効率化する方法として提案された．しかし，その後，上記のように，物理モデルのための方法としても利用されるようになった．

注意：本章では，状態変数は全て観測可能であると仮定する．観測可能でない場合には，状態推定手法と組み合わせる必要がある．

2.3 ハミルトニアンニューラルネットワーク

ニューラル常微分方程式は，多層パーセプトロンの普遍近似性のために，基本的には，どのような関数でも作ることができる．しかし，逆に，どのような関数でも作れてしまうため，特に，物理モデリングに利用する際には，物理現象としてあり得ないような関数を作ってしまうかもしれない．そのようなモデルは，物理学の諸法則を守らないものとなってしまうかもしれず，モデルの信頼性に疑問が残ってしまう．

実際，ニューラル常微分方程式で学習された常微分方程式を数値的に解くことで現象を予測しようとすると，多くの場合，短時間の予測は非常に高精度になるものの，長い時間が経過すると解が発散してしまうといった不都合がおこることが知られている．これは，エネルギー保存則などの物理法則が保たれていないためであると考えられている．

そこで，物理法則を保ちつつ，データ駆動的にモデルを学習する方法として，ハミルトニアンニューラルネットワークという方法が 2019 年に提案された [2]．この方法は，ハミルトン力学の運動方程式であるハミルトン方程式

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} q \\ p \end{pmatrix} = \begin{pmatrix} O & I \\ -I & O \end{pmatrix}\begin{pmatrix} \dfrac{\partial H}{\partial q} \\ \dfrac{\partial H}{\partial p} \end{pmatrix} \tag{4}$$

に基づく．$q\colon t \in R \mapsto q(t) \in R^N$ は一般化座標，$p\colon t \in R \mapsto p(t) \in R^N$ は一般化運動量と呼ばれる．$H\colon (q,p) \in R^N \times R^N \mapsto H(q,p) \in R$ はハミルトニアンと呼ばれ，系の全エネルギーを表す．この方程式はニュートンの運動方程式と等価であることが知られている．また，直接計算によって確認できるように，エネルギー保存則が成り立つ：

$$\frac{\mathrm{d}H}{\mathrm{d}t} = 0. \tag{5}$$

ハミルトニアンニューラルネットワークでは，この方程式に現れるエネルギー関数 H をニューラルネットワーク H_{NN} でモデル化する．具体的には，まず，ニューラル常微分方程式と同様に，状態変数とその微分のデータ $\{(q_1, p_1, \dot{q}_1, \dot{p}_1), \ldots, (q_m, p_m, \dot{q}_m, \dot{p}_m)\}$ が与えられていると仮定し，

$$\sum_{j=1}^{m} \left\| \begin{pmatrix} \dot{q}_j \\ \dot{p}_j \end{pmatrix} - \begin{pmatrix} O & I \\ -I & O \end{pmatrix}\begin{pmatrix} \dfrac{\partial H_{NN}}{\partial q} \\ \dfrac{\partial H_{NN}}{\partial p} \end{pmatrix} \right\|^2 \tag{6}$$

を最小化することでニューラルネットワークを学習させる．この式に現れるニューラルネットワーク H_{NN} の微分は，自動微分と呼ばれるアルゴリズムを用いると計算することができる．

このモデルはハミルトン方程式に基づくモデルであるため，物理学の諸法則が成立する．例えば，自明にエネルギー関数である H_{NN} は保存する：

$$\frac{\mathrm{d}H_{NN}}{\mathrm{d}t} = 0. \tag{7}$$

この性質のため，H_{NN} が真のエネルギーの良い近似になっていれば，H_{NN} が保存することから H も大きく変動することはなく，ニューラルネットワークによる予測が安定すると期待できる．また，これに加えて，ハミルトン方程式のもつ時間可逆性も引き継いでいる．

2.4 散逸系への応用

ハミルトニアンニューラルネットワークは，ハミルトン系を学習するための手法であり，特に，厳密にエネルギー保存則が成り立つような系を対象としている．しかし，例えば，振り子の運動をモデル化したい場合，現実の振り子は摩擦や空気抵抗などを受け，エネルギーは完全には保存しないであろう．そのため，エネルギーが厳密に保存する保存系だけでなく，エネルギーが減衰する散逸系についても考えたくなる．

実は，ハミルトニアンニューラルネットワークは，わずかな修正のみで，散逸系に対して適用することができる．まず，ハミルトン方程式を一般化した

$$\frac{\mathrm{d}u}{\mathrm{d}t} = M(u)\frac{\partial H}{\partial u} \tag{8}$$

の形に表される方程式について考えてみよう．$u\colon t \in R \mapsto u(t) \in R^N$ は状態変数であり，$M(u)$ は u に依存する行列である．$H(u)$ は u に依存する，ハミルトニアンとは限らない，何らかの意味でのエネルギー関数であるとする．簡単な計算で確認できるとおり，$M(u)$ が歪対称行列である場合には，エネルギー保存則が成り立ち，$M(u)$ が半負定値であるときには，エネルギー散逸則が成り立つ．実際，

$$\frac{\mathrm{d}H}{\mathrm{d}t} = \frac{\partial H}{\partial u}^{\top}\frac{\mathrm{d}u}{\mathrm{d}t} = \frac{\partial H}{\partial u}^{\top} M(u)\frac{\partial H}{\partial u} \tag{9}$$

であるが，$M(u)$ が歪対称行列のときには，この量は 0 となり，半負定値であるときには，この量は 0 以下となる．

これを踏まえると，もしも，散逸系を学習したいときには (8) の形の方程式をモデルとして採用し，$H(u)$ だけでなく $M(u)$ もニューラルネットワークを利用してデータから学習すればよい．ただし，$M(u)$ は半負定値にする必要があるが，そのためには，適当な行列 $A_{NN}(u)$ をニューラルネットワークで作成しておき，

$$M(u) \simeq -A_{NN}(u)^{\top}A_{NN}(u) \tag{10}$$

などとすればよい．あるいは，散逸項が小さい摩擦項などであり，ハミルトン方程式から大きく変わらないと期待できるのであれば，

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} q \\ p \end{pmatrix} = \begin{pmatrix} \Lambda_1 & I \\ -I & \Lambda_2 \end{pmatrix}\begin{pmatrix} \dfrac{\partial H_{NN}}{\partial q} \\ \dfrac{\partial H_{NN}}{\partial p} \end{pmatrix} \tag{11}$$

などといったモデルを使うこともできる．Λ_1, Λ_2 は半負定値行列であるが，例えば，単純な摩擦の場合には単位行列の定数倍などとなるので，(10) のような複雑なモデルよりも簡単な形を仮定することもできる．

注意：第 4.2 節で，ニューラルネットワークはベクトル値関数を学習するものと説明したが，ベクトルの成分を行列の形に並べ替えれば，全く同じ関数で，実質的に行列を近似することもできる．

2.5 数値実験例

ここまでの方法について，いくつかの数値実験結果を示しておこう．ハミルトニアンニューラルネットワークとその変種を利用して，保存系，散逸系のそれぞれを学習する実験を行ってみよう．簡単な例として，非線形振動子のモデルであるダフィング方程式

$$\frac{\mathrm{d}^2 q}{\mathrm{d}t^2} = -\alpha q - \beta q^3 - \gamma\frac{\mathrm{d}q}{\mathrm{d}t} \tag{12}$$

を扱う．$\alpha, \beta, \gamma \in R$ はパラメータである．この方程式は，$\gamma = 0$ の場合にハミルトン系となり，エネルギー

$$H(q,p) = \frac{\alpha}{2}q^2 + \frac{\beta}{4}q^4 + \frac{1}{2}p^2 \tag{13}$$

が保存する．ただし，$p = \dfrac{\mathrm{d}q}{\mathrm{d}t}$ である．この系について，ハミルトニアンニューラルネットワークを利用してハミルトニアンを学習する実験を行った．パラメータについては $\alpha = 1, \beta = 2, \gamma = 0$ と設定した．データとしては，初期条件として，$q(0), p(0)$ を標準正規分布で与え，$t = 0$ から $t = 5$ までを 4 次精度の Runge-Kutta 法で数値的に解いた軌道を 100 通り与えた．各軌道について，時間を等間隔に 50 点に区切った各時刻において，$q, p, \dfrac{\mathrm{d}q}{\mathrm{d}t}, \dfrac{\mathrm{d}p}{\mathrm{d}t}$ を与えた．このうちの 80%を学習用のデータとして用い，20%をテスト用のデータとして利用した．ハミルトニアンを近似するためのニューラルネットワークは，3 層の多層パーセプトロンである．隠れ層のサイズは 32 と設定した．学習のための最適化アルゴリズムには Adam アルゴリズムを用い，学習率は 0.001 と設定した．学習のエポック数は 3000 とした．実験には 1 台の T4 GPU を利用したが，この程度の実験であれば，CPU でも十分に計算が可能である．なお，多くの安価な GPU は倍精度計算をサポートしていないため，物理学の実験のためには精度が不十分な場合がある．もしも，状態変数の数が少なく，CPU で十分に動くのであれば，倍精度計算が利用できる CPU を利用する方が好ましいかもしれない．

以上の設定の下で，学習データに対する誤差は最終的に 0.00059 となった．また，テストデータに対する誤差は 0.00092 であった．誤差は平均二乗誤差として計算されているため，テストデータに対する二乗しない誤差は 0.03 程度である．学習されたハミルトニアンから定まるハミルトン方程式を Runge-Kutta 法を利用して数値的に解いた．**図 1** に，ある初期値からの相空間上の真の軌道を表し，**図 2** に学習されたハミルトン方程式を解いた場合の軌道を表す．これらを比較すると，ほとんど同じ軌道が得られていることがわかる．実際，この例では，状態変数は 2 つだけであるため，真の軌道は $H(q,p) = \mathrm{const}$ で定まる曲線上にのる．したがって，もしも，保存量であるハミルトニアンが正しく学習できていれば，学習されたハミルトニアンに基づく予測結果も，同じような曲線の上にのるはずである．そのため，**図 1**，**図 2** は，今回の実験でハミルトンが正確に学習できていることを表していると言える．次に，時間に対して状態変数をプロットしたものを**図 3**，**図 4** に示す．これらは非常に似ているが，よく見ると完全には一致していない．特に，$t = 30$ 付近の様子を観察すると，2 つの図のグラフは少しずれていることが確認できる．**図 1**，**図 2** で比較した軌道はほぼ一致していたものの，テストデータに対する学習誤差は 0.03 程度であった．**図 3**，**図 4** からは，この誤差によって，状態が変化する速度がやや不正確になってしまったことがわかる．

さて，次に，観測誤差が入った場合の様子を見てみよう．全く同様の実験設定で，$q, p, \dfrac{\mathrm{d}q}{\mathrm{d}t}, \dfrac{\mathrm{d}p}{\mathrm{d}t}$ の各データに平均 0，標準偏差 0.05 の正規分布ノイズを加え，同様に学習して

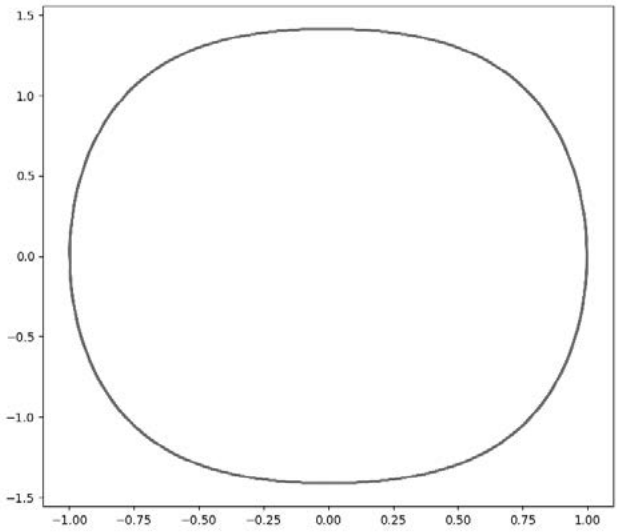

図 1　相空間上のダフィング方程式の真の軌道．

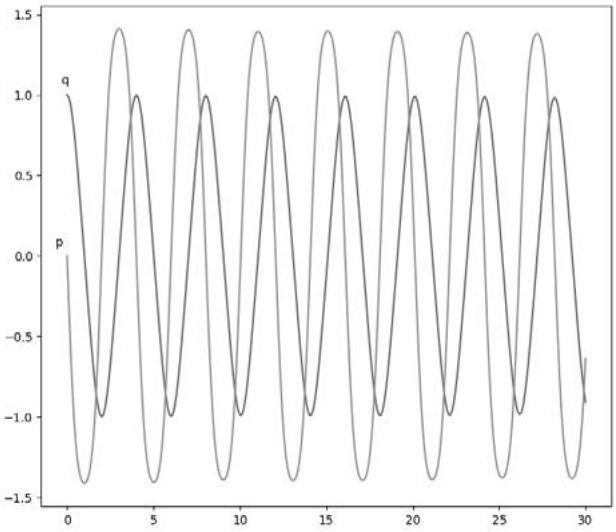

図 3　ダフィング方程式の真の解．

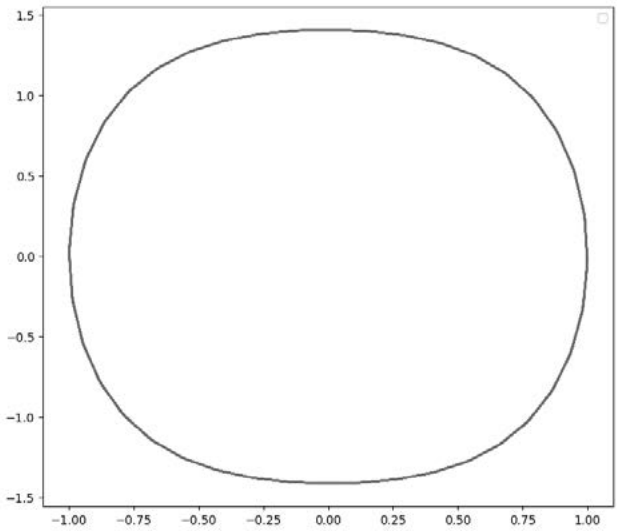

図 2　学習されたハミルトン方程式の解の相空間上の軌道．

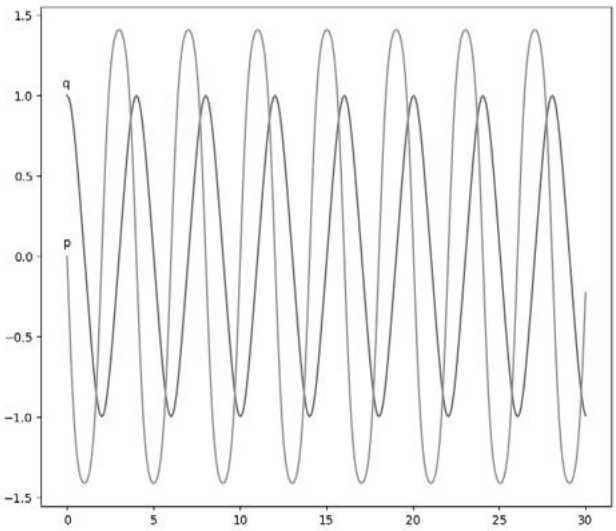

図 4　学習されたハミルトン方程式の解．

みた．今回は学習データに対する誤差は 0.035, テストデータに対する誤差は 0.105 となった．ノイズがない場合に比較して，かなり大きな誤差が入っていることがわかる．**図 5** に学習されたモデルを利用して予測した相空間上の軌道を，**図 6** に各状態変数の時間変化を示す．これらの実験では，ノイズがない場合の実験と同じ初期条件を利用している．ノイズがない場合の結果と比べると，ノイズがあるにも関わらず，かなり正確に予測できているといって良いであろう．**図 7** に，真の解の軌道とノイズを含むデータで学習されたモデルによる予測軌道を，相空間上にプロットしたものを示す．これらを比較すると，少しずれてはいるものの，かなり高い精度で軌道が予測できていることがわかる．

最後に，減衰が入った場合についても，同様の実験を行ってみよう．これまでと全く同じ条件で，減衰の強さを表すパラメータを $\gamma = 0.1$ としてみた．まず，ノイズが無い場合については，学習データに対する誤差は 0.0019, テストデータに対する誤差は 0.0062 となった．減衰がない場合に比較すると誤差は少し大きくなっている．これは，単純にエネルギーを推定するだけでなく，減衰の強さも推定しなくてはいけないため，推定が難しくなったことが原因と思われる．**図 8** に真の軌道を，**図 9** にモデルによって予測された軌道を示す．2 つの軌道を比較すると，モデルによる予測は非常に正確であることがわかる．

次に，平均 0, 標準偏差 0.05 の正規分布ノイズを加えた場合についても，同様に学習してみた．学習データに対する誤差は 0.0215, テストデータに対する誤差は 0.0578 となった．ノイズがない場合に比較して，かなり誤差が大きくなっている．推定された γ の値は 1.024 であった．正確な値は 1.0 であったことを考えると，誤差 2%程度で推定ができていることがわかる．ノイズがない場合と同じ初期値からの軌道を予測した結果を**図 10** に示す．ノイズがない場合と比べても，それほど大きく違わない予測結果が得られた．最後に**図 11** に真の軌道と予測された軌道を相空

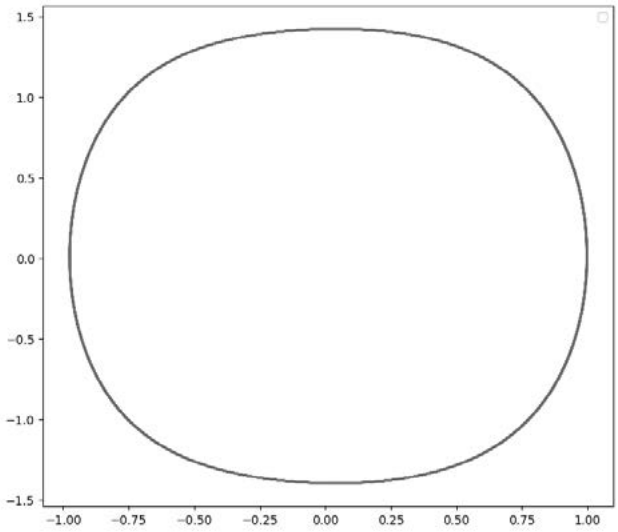

図 5　ノイズが入ったデータから学習された方程式の解の相空間上の軌道．

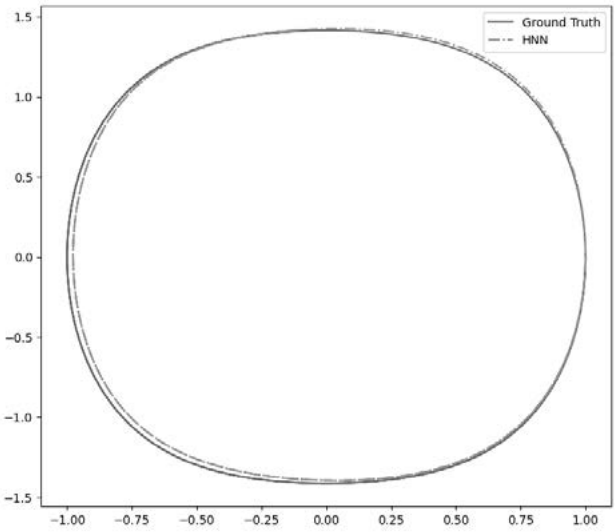

図 7　ノイズが入ったデータから学習されたハミルトン方程式の解と真の解の相空間上の軌道．

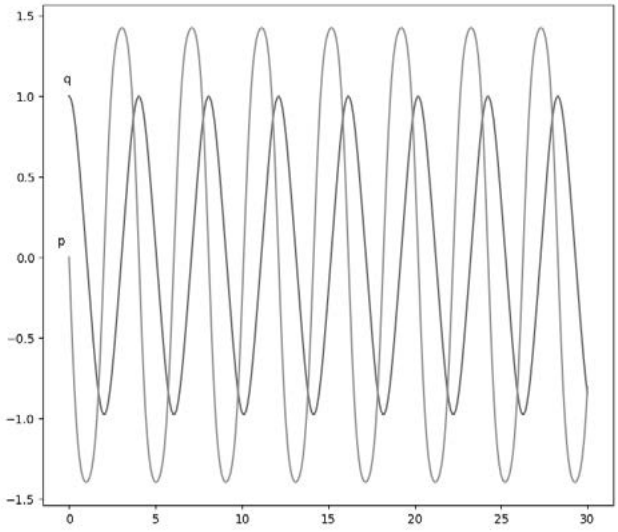

図 6　ノイズが入ったデータから学習されたハミルトン方程式の解．

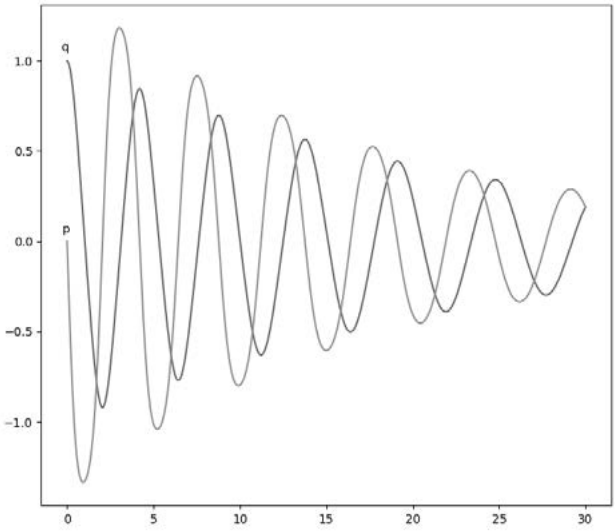

図 8　減衰項をもつ場合の真の軌道．

間上にプロットしたものを示す．ノイズが入ったデータから学習したにもかかわらず，非常に正確な軌道が予測できていることが確認できる．これは，モデルの形として，ハミルトン系やそれに減衰項を加えたものを仮定していることが影響していると考えて良いであろう．観測データにノイズが含まれたとしても，ハミルトン系などの形を仮定することで，平均的な挙動から正しくエネルギー関数が推定でき，結果として，良い予測結果を与えられていることができていると考えられる．

2.6　ラグランジアンニューラルネットワーク

第 4.3 節で紹介したハミルトニアンニューラルネットワークは，エネルギー保存則を保つ方法であるという意味で良い方法になっている．しかし，観測データとして，特に，一般化運動量のデータが必要となっており，これは，実用化を妨げる大きな原因となる．まず，この問題点について説明する．

ハミルトニアンニューラルネットワークでは (4) 式で表されるハミルトン方程式を利用してモデル化する．しかし，この方程式に現れる一般化運動量は，通常，ラグランジアン $L\colon (q,\dot{q}) \in R^N \times R^N \mapsto L(q,\dot{q}) \in R$ を利用して，

$$p = \frac{\partial L}{\partial \dot{q}} \tag{14}$$

のように定義される．そのため，ラグランジアンが既知であれば一般化運動量の式の形もわかるのであるが，ラグランジアンは，一般に，運動エネルギーとポテンシャルエネルギーの差として定義される．すなわち，ラグランジアンが既知である場合には，エネルギー関数も既知である必要がある．ここで，ハミルトニアンニューラルネットワークがエネルギー関数であるハミルトニアンをモデル化しようとしていたことを思い出すと，ハミルトニアンニューラルネットワークを使いたい場面では，エネルギー関数は未知

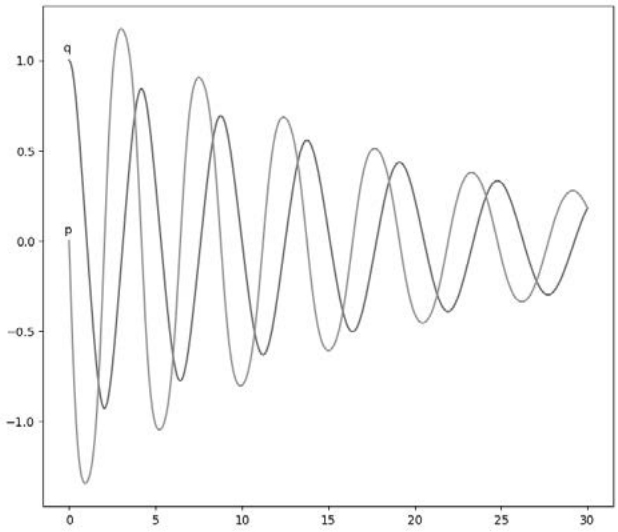

図 9 減衰項をもつ場合の学習された方程式の軌道．

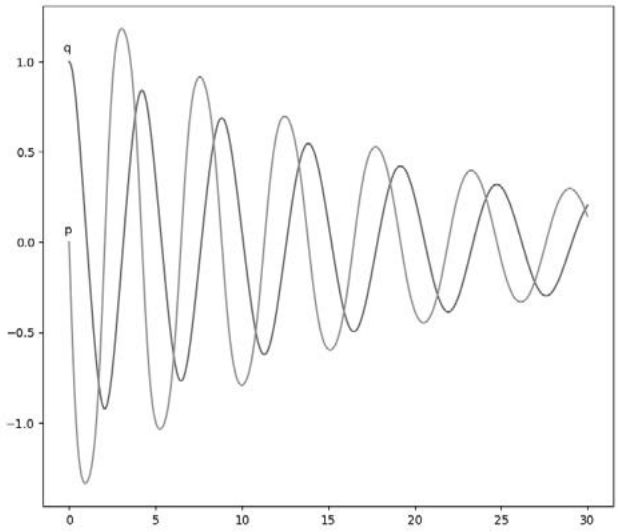

図 10 減衰項をもつ場合に，ノイズが入ったデータから学習された方程式の解．

のはずである．ハミルトニアンニューラルネットワークを学習するためには，データとして p の値が必要なのであるが，エネルギー関数，したがってラグランジアンが未知のため，p がどのような式で表されるかはわからない．そのため，p の値をデータとして用意することができず，このため，ハミルトニアンニューラルネットワークを実データに適用することは難しい．

これを解決するための一つの方法が，ラグランジアンニューラルネットワークである [3]．ラグランジアンニューラルネットワークでは，ハミルトン力学ではなく，ラグランジュ力学を利用してモデルを構築する．ラグランジュ力学の運動方程式はオイラー・ラグランジュ方程式であるが，この方程式は，与えられたラグランジアン $L(q,\dot{q})$ から

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = 0 \tag{15}$$

のように定義される．ラグランジアンニューラルネットワークの基本的なアイデアは (15) 式に現れるラグランジアン L をニューラルネットワーク L_{NN} で学習することである．ハミルトン方程式とは異なり，オイラー・ラグランジュ方程式には一般化運動量が含まれない．そのため，一般化運動量のデータが無くてもラグランジアンの学習が可能となり，したがって，運動方程式が推定できる．すなわち，この方法は，実データに適用可能な方法である．

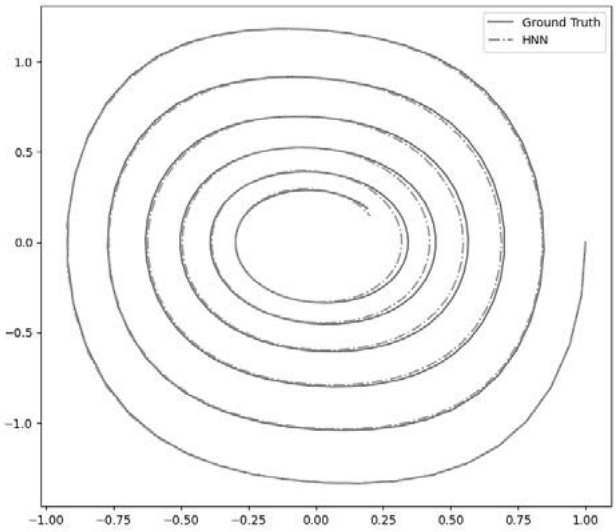

図 11 ノイズが入ったデータから学習された方程式の解と真の解の相空間上の軌道．

実際にデータからラグランジアンを学習する際には (15) 式をそのまま利用することはできない．というのは，(15) 式においてラグランジアン L を定数にすると，L の微分はゼロとなるため，自明にオイラー・ラグランジュ方程式が成立してしまうが，(15) 式の L をニューラルネットワーク L_{NN} に置き換えて素朴に学習すると，実際に L_{NN} は定数関数を学習してしまう．そこで (15) 式を少し書き換えて

$$\frac{\partial^2 L}{\partial \dot{q}^2}\ddot{q} + \frac{\partial^2 L}{\partial q \partial \dot{q}}\dot{q} - \frac{\partial L}{\partial q} = 0 \tag{16}$$

と変形して，$\dfrac{\partial^2 L}{\partial \dot{q}^2}$ が逆行列をもつと仮定して

$$\ddot{q} = \left(\frac{\partial^2 L}{\partial \dot{q}^2}\right)^{-1}\left(-\frac{\partial^2 L}{\partial q \partial \dot{q}}\dot{q} + \frac{\partial L}{\partial q}\right) \tag{17}$$

という式が利用される．具体的には，データとして，$(q,\dot{q},\ddot{q})$ の値が観測されていると仮定し，

$$\sum_{j=1}^{m}\left|\left|\ddot{q}_j + \left(\frac{\partial^2 L_{NN}}{\partial \dot{q}^2}\right)^{-1}\left(\frac{\partial^2 L_{NN}}{\partial q \partial \dot{q}}\dot{q}_j - \frac{\partial L_{NN}}{\partial q}\right)\right|\right|^2 \tag{18}$$

を最小化するように L_{NN} を学習させる．

ラグランジアンニューラルネットワークはオイラー・ラグランジュ方程式に基づくモデルであり，オイラー・ラグランジュ方程式は，ラグランジュ力学の運動方程式であるため，ハミルトニアンニューラルネットワークと同様，エネルギー保存則などの物理法則が成り立つ．例えば，エネルギー保存則として

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L_{NN}}{\partial \dot{q}}\dot{q} - L_{NN}\right) = 0 \tag{19}$$

が成り立つ．

注意：エネルギー保存則以外の保存則，例えば，角運動量保存則などについても同様に成り立つと期待したくなるが，残念ながら，ニューラルネットワークを適切に設計しない限り，エネルギー保存則以外の保存則は期待できない．というのは，ネーターの定理として知られているように，多くの保存則はラグランジアンの対称性に由来している．エネルギー保存則はラグランジアンの時間対称性に由来する性質であり，ニューラルネットワークで表されたラグランジアン $L_{NN}(q,\dot{q})$ は時刻 t を入力としてとらず，したがって t に陽に依存していない．そのため，時間対称性をもつため，エネルギー保存則が成り立つ．しかし，その他の対称性，例えば，回転対称性については，学習されたニューラルネットワークは近似的な対称性はもつかもしれないが，完全な対称性はもたないであろう．そのため，エネルギー保存則以外の対称性から導かれる保存量については，何らかの工夫をしない限り，保存するとは限らない．

2.7 Kolmogorov-Arnold ネットワーク

これまでの手法では，ニューラルネットワークとして，主に多層パーセプトロンを利用してきたが，2024 年ごろに，Kolmogorov-Arnold ネットワーク [4] という，別のタイプのニューラルネットワークが提案され，注目され始めている．このニューラルネットワークも多層パーセプトロンと同様に普遍近似性をもっており，基本的にどのような関数も構成することができる．しかし，多層パーセプトロンと異なり，多変数関数を，単変数関数の合成で関数を表すことができるという大きな特徴をもつ．この特徴のため，多層パーセプトロンに比較して，解釈しやすい関数を学習することができる．

Kolmogorov-Arnold ネットワークは，Kolmogorov-Arnold の表現定理という定理に基づいて設計されている．Kolmogorov-Arnold の表現定理は，元々，ヒルベルトが 1900 年に提示した 23 個の問題のうち，13 番目の問題に関係している．この問題は，7 次以上の代数方程式の解の存在，特に，解が 2 変数の代数関数の合成で書けるかといったことに関する問題である．Kolmogorov-Arnold の表現定理は，この問題の変種である「多変数連続関数を単変数連続関数の合成で表すことができるか」という問題に，肯定的に答えを与えたものである．具体的には，連続関数 $f\colon R^N \to R$ が，単変数連続関数 $\psi_j\colon R\to R$, $\phi_{j,k}\colon R\to R$ の合成で

$$f(x_1,\ldots,x_N) = \sum_{j=1}^{2N+1}\psi_j\left(\sum_{k=1}^{N}\phi_{j,k}(x_k)\right) \tag{20}$$

のように表されることを保証する．

Kolmogorov-Arnold ネットワークは，この定理に基づいて構成されている．具体的には，この定理に現れる $\psi_j\colon R\to R$, $\phi_{j,k}\colon R\to R$ をモデル化することで，関数 f を近似しようとする．これらの関数の近似方法は，基本的に，どのようなものでも構わないが，典型的な方法としては，B-スプライン曲線が用いられる．この曲線を定めるためのパラメータがモデルパラメータとなり，これをデータに合わせて学習する．また，関数を定めるための和の範囲や合成の回数についても，定理の形にとらわれず，自由に変更してよい．

イメージをつかむために，簡単な関数近似問題への適用例を見てみよう．Kolmogorov-Arnold ネットワークのプログラミングのために pykan というパッケージが開発されているのでこれを利用して，簡単な関数をデータから近似してみる．pykan のチュートリアルに示されている例であるが，関数として $f(x,y)=xy$ という関数を近似してみよう．モデルとして，中間層の大きさが 5 である，つまり，5 個の関数を利用した Kolmogorov-Arnold ネットワーク

$$\sum_{j=1}^{5}\left(\psi_j\left(\phi_{1,j}(x)+\phi_{2,j}(y)\right)\right) \tag{21}$$

を利用する．**図 12** は学習されたモデルを可視化したものである．下のほうに表示されている 2 つの頂点は入力である x,y に対応する．上の頂点は出力である．頂点間のグラフは，ネットワークを構成する $\psi_j,\phi_{k,j}$ などの関数を表している．**図 12** の例では，左下の頂点につながっている 2 つのグラフは，どちらも関数 $f(x)=-x$ のように見える．また，その上の方に示されている 2 つのグラフは $f(x)=x^2$ や $f(x)=-x^2$ のような形をしている．一方，右下の頂点からつながっているグラフは，$f(x)=-x$ と $f(x)=x$ のようなグラフに見える．$+$ が示されている頂点は，下から来た入力を足す，という意味であるので，以上をまとめると，学習された関数が

$$(-x-y)^2+(-(-x+y)^2) \tag{22}$$

のような形をしていることがわかる．この式を整理すると

$$(x+y)^2-(x-y)^2 = 4xy \tag{23}$$

となり，確かに関数 xy らしきものが求まっていることがわかる．定数 4 については，上記の考察でグラフの概形から対応する関数を推測したときに係数を無視していたことが原因である．正しく各関数をフィッティングすれば，係数も含めて推定できる．なお，pykan には，推定されたネットワークを簡略化する機能が実装されており，それを利用して無駄な枝を簡略化したものが**図 13** である．これに加え，グラフから数式を推定する機能も実装されており，データから解釈可能な関数を推定するための機能が備わっている．また，この例では，足し算を表す+印の頂点は存在していたが，かけ算を表す頂点は存在していなかった．かけ算を導入すれば，上記のような複雑な形ではなく，関数 xy を直接推定できる．

次に，Kolmogorov-Arnold ネットワークを利用して実装されたラグランジアンニューラルネットワークを試してみよう．データとしては，再びダフィング方程式を利用する．

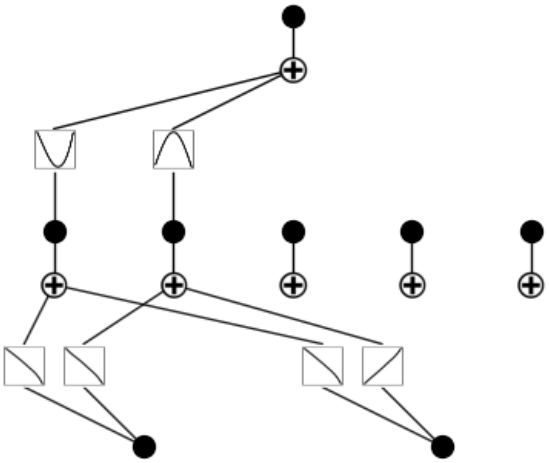

図 12 関数 $f(x,y)=xy$ を学習した Kolmogorov-Aronld ネットワーク．

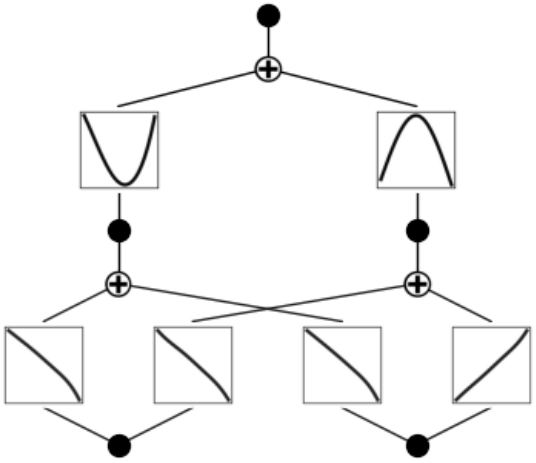

図 13 関数 $f(x,y)=xy$ を学習した Kolmogorov-Aronld ネットワークを簡略化したもの．

ただし，パラメータについては，減衰項がない場合を想定して $\alpha=1/2$, $\beta=1/4$, $\gamma=0$ と設定した．モデル化には 10 個の中間変数をもつ Kolmogorov-Arnold ネットワーク

$$L\left(x,\frac{\mathrm{d}x}{\mathrm{d}t}\right)=\sum_{j=1}^{10}\left(\psi_j\left(\phi_{1,j}(x)+\phi_{2,j}\left(\frac{\mathrm{d}x}{\mathrm{d}t}\right)\right)\right) \quad (24)$$

を利用した．**図 14** に推定された関数の概形を示す．最終的な誤差関数は，平均二乗誤差で 0.0000205 程度であった．正しいエネルギー関数が学習できているかどうかは，やや，わかりにくいが，単純な深層学習手法に比べれば，解釈できる可能性のある結果が得られている．

2.8 おわりに

本章では，主に物理モデリングのための深層学習手法について説明した．本章では，主に方程式を導出する手法を説明したが，この他に，データから方程式だけではなく，対称性や保存則も同時に推定する方法なども提案されている．特に，Kolmogorov-Arnold ネットワークを利用した方法は解釈可能性の点からも，物理モデリングに適している．この分野は急速に発展しているため，近い将来，データから解釈可能な数理モデルを自動的に導出できるようになるかもしれない．

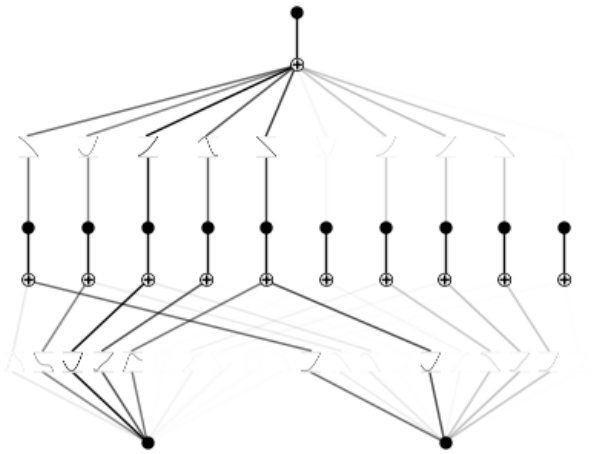

図 14 Kolmogorov-Aronld ネットワークで学習されたラグランジアン．

参 考 文 献

[1] T. Chen *et al.*, Neural Inf Process Syst. **31**, 6572–6583 (2018).

[2] S. Greydanus *et al.*, Neural Inf Process Syst. **32**, 15353–15363 (2019).

[3] M. Cranmer *et al.*, arXiv [cs.LG], available from: http://arxiv.org/abs/2003.04630 (2020)

[4] Z. Liu *et al.*, arXiv [cs.LG], available from: http://arxiv.org/abs/2404.19756 (2024)

谷口（やぐち） 隆晴（たかはる）

神戸大学大学院理学研究科数学専攻教授，理化学研究所革新知能統合研究センター 計算物理機械学習チーム チームディレクター．応用数学・情報学全般の研究に従事．現在は，主に Physics-Informed Neural Networks や作用素学習など，機械学習の物理モデリング・シミュレーションへの応用に取り組んでいる．