Deep Learning による Boltzmann 方程式解析: 制約なし最適化問題への変換

Boltzmann equation analysis by deep learning: Transformation into unconstrained optimization problem

[1]成蹊大，[2]室蘭工大 ○[1]川口 悟，[2]高橋 一弘，[2]大鎌 広，[2]佐藤 孝紀，[1]村上朝之
[1]Seikei Univ. [2]Muroran I.T. ○[1]S. Kawaguchi, [2]K. Takahashi, [2]H. Ohkama, [2]K. Satoh, and [1]T. Murakami
E-mail: skawaguchi@st.seikei.ac.jp

1. はじめに

人工ニューラルネットワーク(Artificial Neural Network: ANN)を応用した微分方程式のメッシュフリー数値解法 Physics-informed Deep Learning (PIDL)が注目されている[1]。本研究の目的は，PIDL を応用することで，位相空間における荷電粒子の連続の式である Boltzmann 方程式を，任意の境界条件の下で高精度に解くことができる汎用数値解法を確立することである。これまで，直流平等電界下における流動平衡状態の電子スオームの電子速度分布関数$f(v_x, v_z)$に関する以下の Boltzmann 方程式を PIDL で解くことが可能であることを示した[2]。

$$\bar{R}_i f(v_x, v_z) + \frac{eE}{m}\frac{\partial f(v_x, v_z)}{\partial v_z} - J_{in}^c f(v_x, v_z) + J_{out}^c f(v_x, v_z) = 0 \quad (1)$$

e: 電気素量, E: 電界強度, m: 電子質量,
$\boldsymbol{v} = (v_x, v_z)$: 電子の速度，$\bar{R}_i$: 実効電離衝突周波数
$J_{in}^c f(v_x, v_z)$, $J_{out}^c f(v_x, v_z)$: 衝突による電子の流入出を表す項

前報[2]では，$f(v_x, v_z)$を ANN で近似し，損失関数 L[(2)式]の値が最小となるよう ANN 内のパラメータを決定(ANN の学習)することで，(1)式の数値解を求めた。

$$L = \frac{1}{N_f}\sum_{i=1}^{N_f}\left[F(v_{x,i}, v_{z,i})\right]^2 + \lambda|C_f - 1| \quad (2)$$

ここで，$F(v_{x,i}, v_{z,i})$は(1)式の左辺，λ は Lagrange の未定乗数，C_fは$f(v_x, v_z)$の規格化定数である。(2)式右辺第 1 項は，ANN が微分方程式をどの程度満たしているかを表す項(L_f)，(2)式右辺第 2 項は，$F(v_{x,i}, v_{z,i}) = 0$を満たす解を一意に定めるための制約条件を表す項(L_b)である。つまり，L の最小化は，制約つき最適化問題を解くことに相当する。

Wang *et al.*[3]は，PIDL における損失関数 L の最小化において，L_f と L_b の最小化の程度に不均衡が生じる場合があることを明らかにし，これが微分方程式の数値解の精度を低下させることを報告している。今後，より高次元または複雑な境界条件下の Boltzmann 方程式を PIDL を応用して解く上で，Wang *et al.*が指摘した問題に直面する可能性がある。そこで我々は，電子の速さ分布の決定を ANN の学習から分離することで，ANN の学習を制約なし最適化問題に変換する，つまり L_b を L から取り除く方法を提案する。提案手法の有効性を Ar ガス中における(1)式の数値計算によって検証したので報告する。

2. 電子の速さ分布の計算法と損失関数の再定義

区間$[0, v_m]$を N_v 等分し，j 番目の電子の速さを v_j とする。v_m は電子の速さの上限値を表し，電子の速さ分布の値が 0 に十分近い値となるように設定する。k epoch における電子の速さ分布 $f_k(v_j)$を次のように更新する。

$$f_1(v_j) = f(v_j)/C_f \quad (3)$$

$$f_{k+1}(v_j) = 0.9 f_k(v_j) + 0.1 f(v_j)/C_f \quad (k > 0) \quad (4)$$

$$f(v) = 2\pi\int_0^{\pi} f(v\sin\theta, v\cos\theta)\sin\theta\, d\theta \quad (5)$$

$v_j < v \leqq v_{j+1}$ における電子の速さ分布 $f_k(v)$を cubic spline によって求める。

損失関数を(6)式のように再定義する。$f(v_x, v_z)$を ANN によって表現し，L の値が最小となるように ANN 内のパラメータを決定する。

$$L = \frac{1}{N_f}\sum_{i=1}^{N_f}\left[F(v_{x,i}, v_{z,i})\right]^2 \quad (6)$$

ここで，$F(v_{x,i}, v_{z,i})$内の$J_{in}^c f(v_x, v_z)$については，$f_k(v)$を使って計算し，その計算値を定数として扱う。L を最小化するためには，$J_{in}^c f(v_x, v_z)$の値に応じて，(1)式を満たすように ANN が表現する$f(v_x, v_z)$が変化しなければならない。このため，L を最小化する過程で，ANN は(1)式を満たす規格化された$f(v_x, v_z)$を表現するものになると予想される。(6)式においては，(2)式における制約条件 $L_b = \lambda|C_f - 1|$が含まれておらず，ANN 内のパラメータの決定問題を制約なし最適化問題に変換できた。また，(4)式のように速さ分布を更新することで，数値解のふらつきを抑えることができ，学習係数のスケジューリングが不要となった。

3. 計算方法および計算条件

Wang *et al.*が提案した ANN を用いる。ANN の中間層の層数を 5 層，各層のニューロン数を 100 とする。中間層と出力層の活性化関数を $x\tanh(\sigma(x))$ (Mish 関数)[4]および $\exp(-x)$とする。ただし，x はニューロンの活性，$\sigma(x) = \log(1 + \exp(x))$である。ANN の学習アルゴリズムとして AMSGrad ($\eta = 0.01$, $\varepsilon = 1.0\times10^{-6}$, $\beta_1 = 0.9$, $\beta_2 = 0.99$)[5]を用いる。計算では，$N_v = 200$, $N_f = 4048$，$v_m = 2.0\times10^7$ m/s とし，$v_{x,i}$, $v_{z,i}$ については一様乱数を用いてサンプリングする。また，学習前に物理量の無次元化を行う[2]。電気学会が推奨する Ar ガスの電子衝突断面積[6]を使用する。

4. 計算結果および考察

Table 1 は，換算電界 1000 Td における Ar ガス中の電子の平均エネルギー$\bar{\varepsilon}$，電子ドリフト速度(flux drift velocity) W_v，$\bar{R}_i$の計算値を Monte Carlo (MC)法による計算値と併せて示す。MC 法による計算値を真値とした場合の相対誤差(Rel. Err.)も Table 1 に示す。今回の計算値は MC 法による計算値と約 0.6%の範囲内で一致した。

今回使用した ANN は，前報[2]のものと比べ，中間層が深く，ニューロン数が少ない。これによって，ANN 内のパラメータ数を前報[2]における ANN と比べ約 1/8 まで削減することができた。一般に，中間層を深くすると，ANN の学習が困難となるが，ANN の構造の変更と ANN の学習を制約なし最適化問題に変換したことが相まって，ANN の学習が進むようになったと考えられる。

参考文献

[1] M. Raissi *et al.*: arXiv.1711.10561 (2017).
[2] 川口ら: 第 67 回応用物理学会春季学術講演会 14p-PB4-10 (2020).
[3] S. Wang *et al.*: arXiv.2001.04536 (2020).
[4] D. Misra: arXiv.1908.08681v2 (2019).
[5] S. J. Reddi *et al.*: arXiv.1904.09237 (2019).
[6] 倉知，中村: 電気学会放電研資 ED-89-72 (1989).

Table 1. Comparison between the electron transport coefficients calculated by the present and Monte Carlo (MC) methods.

	$\bar{\varepsilon}$ (eV)	W_v (10^6 m/s)	$\bar{R}_i$ (10^8 s^{-1})
Present	15.285	5.867	5.358
MC method	15.264	5.905	5.342
Rel. Err. (%)	0.138	-0.642	0.300