精度保証付き数値計算の未来像
—AI時代の計算機援用証明—

田中　一成

数値計算の大規模化が進むなか，コンピュータが返す計算結果の信頼性をいかに保証するかが益々重要な課題となっている．本稿では，丸め誤差や離散化誤差といった代表的な誤差を整理し，区間演算をはじめとする精度保証付き数値計算の基本的な考え方を概説する．さらに，不動点定理やニュートン・カントロビッチの定理を用いた計算機援用証明手法を取り上げ，偏微分方程式の解の存在性や一意性を数学的に保証する方法を紹介する．最後にニューラルネットワーク近似と精度保証を組み合わせたアプローチ“Learn and Verify”を取り上げ，AI技術が精度保証分野にもたらす可能性と課題を展望する．

キーワード：精度保証付き数値計算，計算機援用証明，区間演算，不動点定理，ニュートン・カントロビッチの定理，優解劣解法，深層学習，ニューラルネットワーク

1. はじめに

現代社会において，数値計算は科学技術や工学の幅広い分野で不可欠な基盤技術となっている．気象予測や防災，構造解析・最適化，さらには医療分野や宇宙開発に至るまで，膨大な計算が日常的に行われ，そこから得られる知見が実世界における設計や意思決定に大きく寄与している．計算機の性能は向上し続けており，扱われる問題の規模も年々拡大している．スーパーコンピュータの性能は近年エクサスケール（毎秒 10^{18} オーダー）に到達し，今後さらに発展することも予想される．一方，GPU が数値計算に積極的に活用されるようになっており，一般ユーザーが解くことができる問題の大規模化も進んでいる．

さらに，注目の的となっている大規模言語モデル (LLM) は，計算量の増加に一層拍車をかけると考えられる．すでに多様なタスク（プログラム）を自動実行する AI エージェントが登場しており，科学技術計算の分野でもこれらの技術が導入される可能性が高い．これまで人間がエンターキーを押して実行していたタスクの実行が LLM により代替され，より多くの数値シミュレーションや最適化などの数値計算が LLM の命令により行われるようになると筆者は予想する．こうした自動化と大規模化の進展により，これまで以上に大量かつ複雑な数値計算を迅速に実行できるようになる一方，**コンピュータが弾き出す数値計算の結果は，どこまで正しいのか？** という根本的な問題が，今後いっそう顕在化してくるに違いない．

数値計算には必ず誤差がついて回る．浮動小数点演算における丸め誤差や，反復法を打ち切ることで生じる打ち切り誤差，連続方程式を離散化する際の離散化誤差など，さまざまな誤差が要因となり，ときにはわれわれが想定する「真の解」とは大きくかけ離れた計算結果が導かれることがある．一回あたりの誤差がわずかであっても，大量の反復計算を通じて蓄積すれば最終的な結果に大きな影響を及ぼしかねない．さらに，深刻なエラーや計算結果の逸脱が起こる可能性が低くても，実行回数が増えればその確率は増大する．近年は計算高速化のために半精度や単精度の低精度演算を併用するケースも多く，対策なしにこれらを乱用すると深刻な誤差を招きかねない．特に大規模計算時には，前述のとおりより一層の注意が必要である．エクサスケール時代に突入した今，数値計算の結果検証や再現性確保の重要性が一段と増している．

こうした誤差の問題に対して，**精度保証付き数値計算**と呼ばれる手法が提案されてきた．これは，得られた計算結果に含まれる誤差を厳密に評価し，対象とする問題の真の解がどの範囲に存在するかを数学的に保証する数値計算技術である．本稿では，数値計算における誤差の種類とその影響を概観し，精度保証の必要性を再評価する．続いて，精度保証付き数値計算を実現するための具体的手法や，それを応用した計算機援用証明の手法を紹介する．最後に，近年のニューラルネットワーク (NN) を用いた数値計算手法と関連づけて，筆者の研究“Learn and Verify”アプローチについて触れる．

たなか　かずあき
早稲田大学国際理工学センター
〒 169–8555 東京都新宿区大久保 3–4–1
tanaka@ims.sci.waseda.ac.jp

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

本稿が，読者の皆様が数値計算の誤差や精度保証付き数値計算の基本的な考え方を理解する一助となり，ひいては数値計算の将来を考察するきっかけとなれば幸いである．

2. 数値計算の誤差と精度保証の必要性

現代のほぼすべての CPU や GPU は IEEE 754 と呼ばれる規格に準拠して数値表現や演算を行っている．正式な規格名は IEEE Standard for Floating-Point Arithmetic であり，1985 年に初めて制定され，2019 年に最後の改訂がなされている（2025 年 2 月 9 日現在）．現在 IEEE 754 には五つの数値表現規格があるが，その中でも頻出なのは「単精度浮動小数点数（いわゆる binary32 や single 型）」と「倍精度浮動小数点数（いわゆる binary64 や double 型）」である．以下では，この規格に従う数値計算の過程で発生する代表的な誤差を確認する．

2.1 丸め誤差

有限桁の 2 進数を採用する浮動小数点数では，計算の途中で四捨五入に似た丸め処理が行われる．たとえば 1/3 や π のような無限小数は，コンピュータ上で厳密に表現できないため，できるだけ近い浮動小数点数に丸めて表現される．10 進数では有限桁で表せる小数でも，2 進数では有限桁で表せないこともある．たとえば 8.3 は 2 進数では表せないため，8.3 と入力した時点で誤差（入力誤差）が生じる．四則演算などの演算時にも丸め誤差は繰り返し発生し，極端な例では最終的には非常に大きな誤差が生じる場合がある．

2.1.1 桁落ち

深刻な丸め誤差として有名なのが**桁落ち**である．これは互いに非常に近い値同士を引き算する際に，有効桁が大幅に失われる現象である．以下では，桁落ちの具体例を紹介する．2 次方程式 $ax^2+bx+c=0$ の解の公式

$$x=\frac{-b+\sqrt{b^2-4ac}}{2a} \tag{1}$$

の分子を有理化すると

$$x=\frac{2c}{-b-\sqrt{b^2-4ac}} \tag{2}$$

が得られる．数学的には等価なので，当然両者はいつも同じ値になるはずである．しかし $a=1$，$b=10^{15}$，$c=10^{14}$ を代入して倍精度で計算すると，公式 (1) からは $x=-0.125$ が，公式 (2) からは $x=-0.10000000000000002$ が得られる．このように，数学的には等価なはずの式から 25% も違う値が得られる．なお，真の解は後者に近い $x=-0.1$ である．

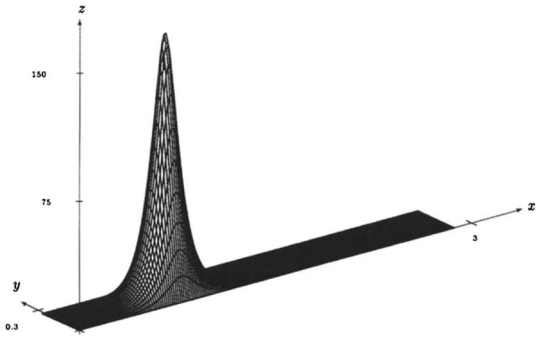

図 1 Emden 方程式 $-\Delta u=u^2$ の同次 Dirichlet 境界値問題の非対称な近似解（文献 [1] の図 6.3 より転載）

この原因は，$b^2=10^{30}$ が $4ac=4\times10^{14}$ よりも遥かに大きく，b と $\sqrt{b^2-4ac}$ が非常に近い値となるため，それらの差をとったときに桁落ちが生じてしまうからである．この例では公式 (2) を用いた際にそのようなことは起きないが，b の符号を逆にすると同様に桁落ちが発生する．式に数値を代入する際には，近い数値同士の引き算を行っていないかをチェックし，式変形や計算順序の変更などの工夫をすることで，桁落ちを回避できることがある．

2.2 離散化誤差

離散化誤差とは，微分方程式などの連続的な問題を離散化して計算機で扱うときに生じる誤差である．たとえば，有限要素法や有限差分法では，空間や時間を有限のメッシュ（格子や三角形要素など）に区切って近似するため，本来の連続解とのずれが生じる．この離散化誤差はメッシュのサイズを小さくすることで一般的には抑えられるが，メッシュ数の増加によって計算量が増大し，同時に丸め誤差も増えるというトレードオフがある．

2.2.1 離散化誤差の例

Breuer-Plum-McKenna は，長方形領域上の Emden 方程式 $-\Delta u=u^2$ に同次 Dirichlet 境界条件（境界上で 0 となる境界条件）を課した数値計算において，図 1 に示すように，離散化誤差による spurious solution（以下「幻影解」と呼ぶ）が得られることを報告した [1]．Gidas-Ni-Nirenberg の理論［2］によりこの問題は非対称な解をもたないことが知られているが，彼らは選点法で図 1 に示す非対称な解を得たのである．なお，この解はスペクトル法や有限要素法などほかの手法でも再現できる．

文献 [3] では，同問題をフーリエ・ガレルキン法で離散化することで有限次元の非線形方程式を導き，そ

 Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

の「有限次元の意味での真の解」として非対称解が得られることを報告している．有限次元の問題としては真の解であるから，幻影解が生じた原因は，離散化誤差が原因であると決定づけられる．

2.3 打ち切り誤差

丸め誤差や離散化誤差のほかに代表的な誤差として，打ち切り誤差が知られる．これは，微分方程式や積分を数値的に解く際，無限回の反復や無限和を有限回で打ち切るために生じる誤差である．たとえばある関数のテイラー展開を途中で打ち切るときの剰余項や，ニュートン法などの反復法を用いる場合に有限ステップで止めることに伴う誤差がこれに相当する．反復回数を増やすと打ち切り誤差は小さくなるが，丸め誤差の影響は逆に大きくなることもある．

2.4 誤差と向き合う

IEEE 754 の制定に貢献した W. M. Kahan は誤差について以下のように述べている [4].

> 数値計算結果と正しい値の間に著しく大きい差があることは極めて稀である．常に心配する必要はないほどには稀である．だが無視できるほどに稀ではない．

実に絶妙な言い回しである．要するに，心配しすぎず数値計算は使うべきであるが，誤差を無視はしてはいけない．特に絶対に間違えてはいけない場面（医療など人命に関わる計算や，宇宙開発など失敗すると莫大な経済的損失を生ずる計算など）ではなおさら注意が必要である．

このような背景から，数値計算の結果に対して何らかの保証を与える必要が生じる場合がある．理想的には，得られた**近似解が真の解からどの程度離れているのか**を定量的に把握したい．これを実現する手法の一つが，精度保証付き数値計算である．

3. 精度保証付き数値計算

ここでは，数値計算で生じる誤差を厳密に評価し，数学的に正しい結果を保証する「精度保証付き数値計算」の基本的な考え方を紹介する．なお「精度保証」とは必ずしも高精度な結果を約束するという意味ではなく，誤差の上限や解の包含範囲を明示することで数学的正しさを担保しているという点に注意してほしい．

3.1 区間演算

区間演算とは，対象の区間包含を考えることで誤差の影響を適切に評価し，自動的に演算を行う技術である．特にここでは，実数に対する区間演算を考える．たとえば，実数 x が正確に表現できない状況でも

$$x \in [a, b]$$

という形で実数 x が必ず $[a, b]$ という区間に含まれるということを保証する．区間演算の起源は，1956 年 2 月 29 日付けの須永照雄の手書きの修士論文まで遡る [5]. 1958 年に須永自身がこの内容をもとにした論文 [6] を出版している．その後，1966 年に R. E. Moore が区間演算に関する著書 [7] を出版し，区間演算の理論的基盤を確立した文献として広く認知された．

区間 $X = [a, b], Y = [c, d]$ に対する四則演算は以下のように導かれる．

$$\begin{aligned} X + Y &= [a + c,\ b + d] \\ X - Y &= [a - d,\ b - c] \\ X \times Y &= [\min\{ac, ad, bc, bd\}, \\ &\qquad \max\{ac, ad, bc, bd\}] \\ X \div Y &= X \times [1/d, 1/c] \end{aligned}$$

実際にコンピュータで区間演算を実現するには，区間の下（上）端を計算するときは下（上）向き丸めを用いるよう注意しなければならない．IEEE 754 は丸め方向の変更についても規定していて，多くのコンピュータで実現可能である．また，除算の Y に 0 が含まれる場合は，演算結果として $[-\infty, \infty]$ などの無限区間を許すのが一般的である．三角関数，べき乗，対数などの数学関数に対する区間演算は必ずしも自明ではないが，それらを実装した区間演算ライブラリは数多く開発されている．たとえば INTLAB [8], kv ライブラリ [9], JuliaIntervals [10] などが挙げられる．また，JuliaIntervals を用いて Jupyter Notebook 上で精度保証付き数値計算を実行した実例が文献 [11] で公開されている．

3.1.1 区間は何を包含しているのか

区間演算を利用するにあたって，区間が何を包含しているのかを常に明確化することが必要である．たとえば区間 $[-1, 1]$ に対して

$$[-1, 1]x, \quad x \geq 0$$

はどう解釈すべきだろうか．一見直線にも見えるがその意味は場合により異なる．$[-1, 1]$ が一つの固定された実数（たとえば 0.5）を包含した結果であれば，$[-1, 1]x$ は $x \geq 0$ における「直線を包含した帯」である．しかし，$[-1, 1]$ が $\sin x$ を包含した結果であれば，$[-1, 1]x$ は直線でなく「曲線を包含した帯」である．このよう

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

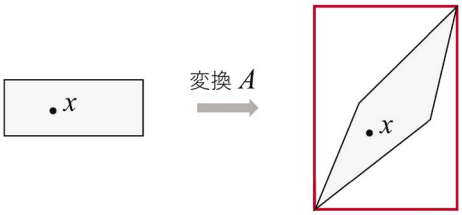

図 2　Wrapping Effect の概念図

に，区間が何を包含したものであるかを常に明確にする必要がある．特に「包含されている数や関数がどの変数に依存するか」という情報は重要になることが多い．後述する計算機援用証明では，区間をその中に値域をとるあるクラス（たとえば連続関数）の集合と捉えることがある．

3.1.2　区間演算の課題：Wrapping Effect

図 2 に示すような多次元ベクトル空間上での区間演算における区間幅増大は Wrapping Effect と呼ばれる．たとえば，区間の直積集合（長方形）

$$x \in [a,b] \times [c,d]$$

を行列 A で変換する場合を考える．変換結果 Ax は長方形以外の形状になり得るが，区間演算ではこれを包含するために，可能な限り小さい長方形で評価せざるを得ない．このように，区間演算は常に安全側に立ち，区間を大きめに見積もるため，演算を繰り返すほど区間幅が増大する傾向にある．また，計算コストも通常の浮動小数点演算に比べると大きい．したがって，**区間演算は必要な箇所に絞って使用する**という考え方が，精度保証付き数値計算の主流となっている．

3.2　精度保証の実用例：連立 1 次方程式

連立 1 次方程式を解くガウスの消去法のアルゴリズム全体を区間演算に置き換えた場合，計算結果自体は常に数学的に正しい．しかし，丸め誤差の蓄積や Wrapping Effect により，行列サイズが数十程度でも区間幅が著しく増大し，実用的な結果が得られなくなってしまう．そこで，区間演算だけに頼らず，連立 1 次方程式の解包含を効果的に得る手法の一例を以下に示す．

Theorem 1（山本 [12]）．A を $n \times n$ の実正方行列，b を $n \times 1$ の実ベクトルとする．さらに R を A^{-1} の近似とし，近似解 $\tilde{x}$ の残差を $r = b - A\tilde{x}$ とおく．このとき

$$\|RA - I\|_\infty < 1 \tag{3}$$

が成り立つならば A は正則であり，真の解 $x^* = A^{-1}b$ に対して以下の不等式が成り立つ：

$$|x^* - \tilde{x}| \le |Rr| + \frac{\|Rr\|_\infty}{1 - \|RA - I\|_\infty} |RA - I| e \tag{4}$$

ここで，$|\cdot|$ は行列（ベクトル）の成分ごとの絶対値を並べた行列（ベクトル）を表す．また，$\|\cdot\|_\infty$ はベクトルの無限大ノルムから誘導される行列の作用素ノルムであり，$\|A\|_\infty = \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} |a_{ij}|$ で与えられる．さらに，$e = [1, 1, \ldots, 1]^{\mathsf{T}}$ である．

この定理を用いるうえで重要なことは，R や $\tilde{x}$ の計算は従来どおり近似的に行ってよいということである．すなわち近似的に得た R と $\tilde{x}$ に対して，条件 (3) と (4) を区間演算で検証すればよい．なお，実際には「条件 (3) の左辺」と「条件 (4) の右辺」の上からの評価だけを行えばよいので，常に上からの評価が得られるよう丸めモードを制御することにより，単純に区間演算を使用するよりもさらに計算コストを削減することが可能である．この工夫により，より実用的で大きなサイズの問題の解を精度保証付きで求めることが可能になる．

精度保証付き数値計算は連立 1 次方程式以外にも，固有値問題，最適化問題，計算幾何学，数値積分，微分方程式など幅広い分野に応用されている [3, 13]．

4.　計算機援用証明

本節では，精度保証付き数値計算を応用した，**計算機援用証明** (Computer-Assisted Proofs) について概説する．誤差の把握だけでなく，対象問題の解が厳密に存在することを同時に証明するものであり，数学的観点から「数値的検証法」や「計算機援用存在証明」とも呼ばれる．また，多くの場合「局所一意性証明」や「非退化性証明」など，存在性以外の重要な性質も保証できる．以下では，主に偏微分方程式を対象とする計算機援用証明の理論を紹介する．ここで取り上げる理論と，前節で概説した区間演算などの精度保証技術を組み合わせることで，計算機援用証明が実現できる．

なお，計算機援用証明は一つの固定された方程式だけでなく，領域や係数などのパラメータをもつ方程式に対しても有効であり，それらを変化させた分岐構造解析に応用されることがある．この場合，特に**計算機援用解析**と呼ぶことがある．

4.1　不動点定理

多くの計算機援用証明においては，不動点定理が用

 Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

いられる．偏微分方程式など無限次元の問題の場合はSchauderの（あるいはBanachの）不動点定理が使用されることが多い．すなわち，問題をコンパクト作用素 F の不動点 $u = F(u)$ の形に持ち込み，縮小性

$$F(X) \subset X$$

を精度保証付き数値計算を援用的に用いて検証する．このアプローチに有限要素法の射影誤差評価を組み合わせることで微分方程式に対する計算機援用証明を実行した事例として，1990年前後の中尾充宏による先駆的な研究がある．この手法の特徴は，真の解が属する無限次元空間からその有限次元部分空間（有限要素空間）への射影 P_h を考え，以下のように不動点方程式を無限次元部分と有限次元部分に分けて縮小性を検証していることである．

$$\begin{cases} P_h u = P_h F(u) \\ (I - P_h)u = (I - P_h)F(u) \end{cases} \tag{5}$$

ここで h は有限要素法におけるメッシュサイズなどの近似パラメータを表す．一つ目の方程式は有限次元であるから，直接的な計算手法で縮小性を検証できる．二つ目の方程式に関する縮小性を検証するには，射影 P_h に関する「補完誤差定数」と呼ばれるノルム評価を与える定数の評価が本質的であり，これは連続と離散のギャップを定量化しているものである．詳細は文献[3]や文献[14]などの文献に記載されている．その後，この手法は発展し，定常Navier–Stokes方程式，Kolmogorov問題，熱対流問題などさまざまな問題に応用されてきた．

4.2 ニュートン・カントロビッチの定理

ニュートン・カントロビッチの定理（以下「NK定理」と呼ぶ）は，ニュートン法の収束や収束範囲を保証する定理である．この定理は一般にBanach空間でも成立するため，偏微分方程式に対する計算機援用証明に使用される．以下では X, Y をBanach空間とし，非線形作用素 $F : X \to Y$ を考える．さらに，F の $v \in X$ におけるフレッシェ微分を F'_v と表記する．

Theorem 2（ニュートン・カントロビッチの定理）**.** $\hat{u} \in X$ を $F(u) = 0$ の解 u の近似とする．次の不等式を満たす $\alpha > 0$ が存在する．

$$\|F'^{-1}_{\hat{u}} F(\hat{u})\|_X \leq \alpha \tag{6}$$

次に $D = B(\hat{u}, 2\alpha + \delta)$ は開球とし，$\delta > 0$ は任意に小さくとれるとする．任意の $v, w \in D$ に対して，次の不等式を満たす $\beta > 0$ 存在すると仮定する．

$$\|F'^{-1}_{\hat{u}}(F'_v - F'_w)\|_{\mathcal{L}(X,X)} \leq \beta \|v - w\|_X \tag{7}$$

このとき

$$\alpha\beta \leq \frac{1}{2}$$

が成り立つならば，$B(\hat{u}, \rho)$ 内に $F(u) = 0$ を満たす解 $u \in X$ が存在する．ただし

$$\rho = \frac{1 - \sqrt{1 - 2\alpha\beta}}{\beta}$$

である．さらにこの u は D 内で一意である．

以上の定理は一見，非常にややこしく見えるかも知れないが，不等式(6)の左辺は**ニュートン法の修正量**のノルムを表している．これがある程度小さいことは，初期値 $\hat{u}$ が真の解に近いことを意味し，ニュートン法が収束しやすくなることを意味する．一方，不等式(7)は問題の難しさそのものに依存する条件であり，F の性質を強く反映する．NK定理には形の似たさまざまなバージョン（以下「NK型定理」と呼ぶ）が存在し，流儀や目的によって使い分けられている．また，実用上は逆作用素ノルム $\|F'^{-1}_{\hat{u}}\|_{\mathcal{L}(Y,X)}$ の評価が上記定理を応用するうえで本質的であり，このノルムの評価をテーマにした複数の論文が出版されている．逆作用素ノルム評価の重要性は先の不動点定理を用いるアプローチにおいても共通するのである．NK定理やNK型定理とその応用については文献[13]の第6章，文献[14]のPart IIなどで詳しく解説されている

4.2.1 近年の計算機援用証明の例

ここでは，NK定理やそれに類するNK型定理を用いた近年の計算機援用証明の事例を三つ紹介する．文献[15]では3DNavier–Stokes方程式の定常解の存在を計算機援用証明している．特に，Scott–Vogelius有限要素空間上でのハイパーサークル法を用いた誤差評価を用いてNK定理の成立条件を検証している．文献[16]では，Suarez–Schopfによる遅延項を含むエルニーニョ方程式に対して，NK型定理を適用している．特に，漸近優ブロック対角行列の性質を利用して同定理の成立条件を確認することで，エルニーニョ現象とラニーニャ現象が交互に現れる周期解が存在することを示している点が特徴的である．文献[17]はパターン形成でよく知られるSwift–Hohenberg方程式を対象にNK型定理を適用している．空間的に局在した非自明なパターンの存在を計算機を援用して厳密に証明している．

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

4.3 その他の計算機援用証明手法

上記以外にも，半群理論を用いた手法 [18] や，優解劣解法に基づく手法 [19] などがあり，独自の特徴をもっている．また，常微分方程式に対しては偏微分方程式とは別のアプローチを取ることができ，たとえば文献 [13] の第 7 章が参考になるだろう．また，力学系分野への応用は文献 [20] などで詳しく解説されている．これらの紹介は紙面の都合上割愛するが，優解劣解法については次節で触れる．

5. ニューラルネットワークと精度保証

深層学習の発展に伴い，ニューラルネットワーク (NN) を数値計算に応用する研究も盛んになってきている．たとえば Physics-Informed Neural Networks (PINNs) は，偏微分方程式 (PDE) などのモデル方程式や境界条件を満たすよう損失関数を設計し，観測データとのバランスを取りながら解を学習する．さらに，単に解を求めるだけでなく，物理的意味をもつ係数の推定にも応用可能である．ほかにも，解写像そのものを学習するニューラルオペレーターやニューラル PDE ソルバーなど，さまざまな方向で数値計算と機械学習の融合が進んでいる．これらの NN ベースの手法には有限要素法などでは扱いにくい高次元問題や境界形状が複雑な問題にも対応しやすいという利点がある．一方で，どこまで厳密な信頼性を保証できるかという課題が指摘されているのも事実である．実際，NN ベースのスキームは学習の初期値やハイパーパラメータに非常に敏感であり，厳密な誤差評価はおろか，有限要素法のような収束定理も未確立な場合が多い．そこで本稿では，近年筆者が取り組んでいる，**NN で微分方程式の解を包み込む**という“Learn and Verify”アプローチの一端を紹介する．

5.1 初期値問題に対する優解劣解法

本節では，筆者の手法を簡潔に示すために，以下の常微分方程式の初期値問題を取り上げる．

$$\begin{cases} \dfrac{du}{dt}(t) = f(t, u(t)), \quad t \in [0, T], \\ u(0) = a \end{cases} \tag{8}$$

ここで，T は正定数，a は実数初期値，$f : [0, T] \times \mathbb{R} \to \mathbb{R}$ は連続関数とする．以下では，優解と劣解を用いて真の解を包含する手法を示す．

Theorem 3. $0 = t_0 < t_1 < \cdots < t_n < t_{n+1} = T$ として区間分割を与える．関数 $\underline{u}, \overline{u} \in \bigcap_{i=0}^{n} C^1([t_i, t_{i+1}])$ が以下の不等式を満たすとする．

$$\begin{cases} \dfrac{d\underline{u}}{dt}(t) \le f(t, \underline{u}(t)), \quad t \in \displaystyle\bigcup_{i=0}^{n} [t_i, t_{i+1}], \\ \underline{u}(0) \le a \end{cases} \tag{9}$$

$$\begin{cases} \dfrac{d\overline{u}}{dt}(t) \ge f(t, \overline{u}(t)), \quad t \in \displaystyle\bigcup_{i=0}^{n} [t_i, t_{i+1}], \\ \overline{u}(0) \ge a \end{cases} \tag{10}$$

すべての $t \in [0, T]$ に対して $\underline{u}(t) \le \overline{u}(t)$ が成立するとき，初期値問題 (8) の解 $u \in C([0, T])$ が存在し

$$\underline{u}(t) \le u(t) \le \overline{u}(t)$$

がすべての $t \in [0, T]$ に対して成り立つ．

不等式 (9) および (10) を満たす関数 $\underline{u}$ と $\overline{u}$ を，それぞれ初期値問題 (8) の劣解，優解と呼ぶ．なお，初期値問題以外のほかの方程式（2 点境界値問題や楕円型偏微分方程式，放物型偏微分方程式など）でも，方程式の「$=$」部分を不等号に変更することにより優解・劣解が定義されるのが一般的である（たとえば文献 [19, 21] などを参照）．

5.2 NN による計算機援用証明

不等式 (9) および (10) を満たす $\underline{u}, \overline{u}$ の構築を NN モデルを用いて行う．ここでは NN は全結合と滑らかな活性化関数からなると仮定し，その重みやバイアスなどのパラメータを動かしてできる関数全体の集合を $\mathcal{A} \subset C([0, T]) \cap C^1((0, T))$ と表す．$\mathcal{A}$ は実際には NN の構造に依存するが，ここでは表記上は区別しない．優解・劣解を学習するためのコスト関数を設計するにあたり，以下では近似解 $\hat{u}$ を用いたアプローチを紹介する．この $\hat{u}$ はたとえば PINNs などの NN ベースの手法で得られたものを想定し，ここでは簡単のため C^1 級であるとする．

本手法で工夫する点の一つは，$\underline{u}$ と $\overline{u}$ を直接一つの NN で表現するのではなく以下のように構成することである．

$$\underline{u} = \hat{u} - v, \quad \overline{u} = \hat{u} + w$$

すなわち $v, w \in \mathcal{A}$ は誤差の役割を担い，この**誤差の部分を NN で学習する**というアプローチを取る．さらに，誤差範囲を NN の構造に直接組み込む工夫も行う．具体的には，NN の出力層にユーザーが望む許容誤差 ε とシグモイド関数[1] $\sigma(x)$ の積，すなわち $\varepsilon\sigma(x)$ を設けることで，v と w の値を 0 と ε の間に制限する．このように構成する最大の利点は，コスト関数に

[1] 0 から 1 の間の値を取る滑らかな関数であればなんでもよいが，学習の安定性を鑑みて単調性があることが望ましい．

 Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

ペナルティ項を加えなくても，v と w が所望の範囲内に収まるため，学習効率が高まることである．なお，$\hat{u}$,v,w を一つの NN モデルで同時に学習することも可能だが，本稿では“Learn and Verify”アプローチの一環としてまず $\hat{u}$ を学習し，その後，固定した $\hat{u}$ に対して v と w を学習する手順を採用している．

5.2.1 優解・劣解学習のためのコスト関数

優解・劣解を学習するためのコスト関数を以下のように定義する．

$$L(\underline{u},\overline{u}) = \underset{\mathcal{S},c_1,c_2}{\mathrm{DSM}}\left[\frac{d\underline{u}}{dt}(t) - f(t,\underline{u}(t))\right] + \underset{\mathcal{S},c_1,c_2}{\mathrm{DSM}}\left[f(t,\overline{u}(t)) - \frac{d\overline{u}}{dt}(t)\right] \tag{11}$$

ここで，DSM(Doubly Smoothed Maximum) はパラメータ c_1, c_2 により定められる softplus 関数と log-sum-exp 関数の合成関数である．具体的には，関数 $g : \mathbb{R} \to \mathbb{R}^n$ に対して

$$\underset{\mathcal{S},c_1,c_2}{\mathrm{DSM}}\left[g(t)\right] = c_2 \log\left(\sum_{t\in\mathcal{S}}\left(1+\exp\left(\frac{g(t)}{c_1}\right)\right)^{\frac{c_1}{c_2}}\right) \tag{12}$$

と定義する．ここで，

- $\mathcal{S}$ は領域 Ω 内のサンプリング点の集合，
- c_1 は softplus 関数の滑らかさを制御するパラメータ，
- c_2 は log-sum-exp 関数の滑らかさを制御するパラメータ

である．max 関数そのものではなく，滑らかにした DSM 関数を用いる理由は，安定性や収束速度の観点から学習上の利点があるためである．実際の実装では，DSM の計算過程でオーバーフローを防ぐ工夫や，コスト関数への安定化項の追加などを行う．

コスト関数 (11) の設計において強調されるべきことは，もし $\hat{u}$ が初期条件を厳密に満たす場合，**優解・劣解の初期条件のペナルティ項を設定する必要がない**点である．これは，前述の v と w の構造により「常に非負」という制約を自動的に満たす仕組みを組み込んであるからこそ実現できる利点である．

5.3 数値例

初期値問題 (8) に対して，以下の設定を考える．

$$f(t) := r(t)u(t)\left(1-\frac{u(t)}{k(t)}\right)$$
$$a = 1, \quad T = 10$$

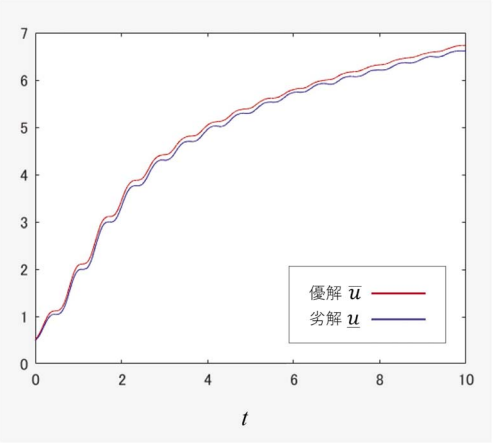

図 3 優解・劣解学習の結果
誤差上限パラメータを $\varepsilon = 2^{-5}$ に設定しているため，優解と劣解の最大差 は ε の 2 倍に当たる 2^{-4} となる．

ただし $r(t) = 2(1+\sin(10t)), k(t) = 2(\log(1+t)+1)$ である．図 3 に，本問題の優解・劣解を NN で学習した結果を示した．NN の構造として，層数 5，各層のニューロン数 30 を採用し，誤差上限のパラメータを $\varepsilon = 2^{-5}$ に設定している．学習後，Theorem 3 の条件を精度保証付き数値計算で検証することにより，得られた優解・劣解の間に真の解が厳密に含まれていることを証明した．

5.4 課題と展望

NN ベースの手法に共通する課題として，ハイパーパラメータの選択が挙げられる．NN は初期値や学習率，層の深さなど，多様な設定に敏感な一方で，最適なパラメータを導くための統一的な指針は十分に確立されていない．さらに，ユーザーが「学習→精度保証→再学習」というサイクルを円滑に回すには，ライブラリや可視化ツールの充実が不可欠である．また，本手法は偏微分方程式にも応用可能だが，常微分方程式であっても連立された方程式系への適用には課題が残る．現時点では，理論面や実装面の両方で十分な成果が得られておらず，今後の重要な研究テーマとして取り組む必要がある．

以上のように，依然として解決すべき課題は少なくないが，深層学習の柔軟性と精度保証技術の融合は，科学技術計算における新たな可能性を切り開く有望な方向性だと考えている．筆者はこのアプローチを“Learn and Verify”と呼ぶ．まず深層学習によって近似解（予測解）を得てから，その結果を事後的に厳密検証 (Verify) することは有力な問題解決アプローチになると思われる．このアプローチは従来の数値計算では扱いに

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.

くかった複雑な問題にも適用しやすい潜在力をもち，精度保証付き数値計算の適用範囲を大きく広げる可能性があると期待される．

6. おわりに

数値計算の大規模化は，今後さらに加速すると考えられる．2024 年に Microsoft と Atom Computing が，1,000 以上の物理量子ビットと 50 の論理量子ビットをもつ中性原子ベースの量子コンピュータの受注を開始したと発表した．現時点では量子優位性の実現には至っていないものの，将来は量子コンピュータが科学技術計算のアクセラレーターとして活躍する可能性がある．もしそうなれば，これまでのノイマン型コンピュータとは比べものにならないほどの計算能力が実現し，計算結果の品質保証がより一層求められる時代になるだろう．一方，LLM が数値計算プログラムを容易に自動実行する時代もそう遠くないと筆者は確信している．これは計算機援用証明の分野にとって，大きなチャンスとなり得る．たとえば，係数パラメータを含む方程式に対し，パラメータ区間を分割して計算機援用解析を行う場合，これまでは分岐点付近でアルゴリズムを切り替えるなどのヒューリスティックな判断が求められた．しかし，LLM がある程度の方針を提案し，証明の実行自体はこれまでどおり手続き的な精度保証プログラムで行うようにすれば，数学的厳密性を維持しつつ効率的な計算機援用解析を実現できるかもしれない．最後に，精度保証付き数値計算や計算機援用証明の基礎を日本語で体系的に学べる文献として書籍 [3, 13] を改めて紹介しておきたい．

謝辞　本稿執筆にあたり助言をいただいた柏木雅英氏（早稲田大学），高安亮紀氏（筑波大学）に深く感謝申し上げる．また，本特集への執筆機会を与えてくださったオーガナイザーの相原研輔氏（東京都市大学）ならびに改訂過程で貴重なご指摘をくださった担当編集委員の伊豆永洋一氏（九州大学）に厚く御礼申し上げる．

参考文献

[1] B. Breuer, M. Plum and P. McKenna, “Inclusions and existence proofs for solutions of a nonlinear boundary value problem by spectral numerical methods,” *Topics in Numerical Analysis*, P. G. Ciarlet and A. Iserles (eds.), Springer, pp. 61–77, 2001.

[2] B. Gidas, W.-M. Ni and L. Nirenberg, “Symmetry and related properties via the maximum principle,” *Communications in Mathematical Physics*, **68**, pp. 209–243, 1979.

[3] 中尾充宏，渡部善隆，『実例で学ぶ精度保証付き数値計算』，サイエンス社，2011.

[4] W. M. Kahan, “The regrettable failure of automated error analysis,” *A Mini-Course Prepared for the Conference at MIT on Computers and Mathematics*, 1989.

[5] T. Sunaga, “Geometry of Numerals,” Master Thesis, The University of Tokyo, 1956.

[6] T. Sunaga, “Theory of interval algebra and its application to numerical analysis,” *RAAG Memoirs*, **2**, pp. 547–564, 1958.

[7] R. E. Moore, *Interval Analysis*, Prentice-Hall, 1966.

[8] S. M. Rump, “INTLAB – INTerval LABoratory,” *Developments in Reliable Computing*, T. Csendes (ed.), Kluwer Academic Publishers, pp. 77–104, 1999.

[9] 柏木雅英，「kv - C++ による精度保証付き数値計算ライブラリ」，http://verifiedby.me/kv/, https://github.com/mskashi/kv （2025 年 2 月 9 日閲覧）

[10] D. P. Sanders and L. Benet, “IntervalArithmetic.jl,” https://github.com/JuliaIntervals/IntervalArithmetic.jl （2025 年 2 月 9 日閲覧）

[11] 高安亮紀，「精度保証付き数値計算のための Julia のパッケージ」，高安研究室ブログ，https://taklab-blog.blogspot.com/2021/01/rigorous-numerics-julia.html （2025 年 2 月 9 日閲覧）

[12] T. Yamamoto, “Error bounds for approximate solutions of systems of equations,” *Japan Journal of Applied Mathematics*, **1**, pp. 157–171, 1984.

[13] 大石進一（編），『精度保証付き数値計算の基礎』，コロナ社，2018.

[14] M. T. Nakao, M. Plum and Y. Watanabe, *Numerical Verification Methods and Computer-Assisted Proofs for Partial Differential Equations*, Springer, 2019.

[15] X. Liu, M. T. Nakao and S. Oishi, “Computer-assisted proof for the stationary solution existence of the Navier–Stokes equation over 3D domains,” *Communications in Nonlinear Science and Numerical Simulation*, **108**, 106223, 2022.

[16] 三浦悠希，大石進一，関根晃太，“連立遅延エルニーニョ方程式における分数調和解の存在の計算機援用証明，”日本応用数理学会論文誌，**34**, pp. 1–18, 2024.

[17] M. Cadiot, J.-P. Lessard and J.-C. Nave, “Stationary non-radial localized patterns in the planar Swift-Hohenberg PDE: Constructive proofs of existence,” *Journal of Differential Equations*, **414**, pp. 555–608, 2025.

[18] M. Mizuguchi, A. Takayasu, T. Kubo and S. Oishi, “A method of verified computations for solutions to semilinear parabolic equations using semigroup theory,” *SIAM Journal on Numerical Analysis*, **55**, pp. 980–1001, 2017.

[19] Y. Matsushima, K. Tanaka and S. Oishi, “Numerical verification for positive solutions of Allen–Cahn equation using sub- and super-solution method,” *Journal of Advanced Simulation in Science and Engineering*, **7**, pp. 136–150, 2020.

[20] J. B. van den Berg and J.-P. Lessard (eds.), *Rigorous Numerics in Dynamics*, *Proceedings of Symposia in Applied Mathematics*, **74**, American Mathematical Society, 2018.

[21] M. S. Carl, V. K. Le and D. Motreanu, *Nonsmooth Variational Problems and Their Inequalities: Comparison Principles and Applications*, Springer, 2007.

 Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.