グラフニューラルネットワークを用いたインテリジェントメッシュスムージング手法の提案

Zhichao Wang[1,2], Xinhai Chen[1,2*], Junjun Yan[1,2], Jie Liu[1,2]

[1*]Science and Technology on Parallel and Distributed Processing Laboratory, National University of Defense Technology, Changsha, 410073, China.
[2]Laboratory of Digitizing Software for Frontier Equipment, National University of Defense Technology, Changsha, 410073.

*Corresponding author(s). E-mail(s): chenxinhai16@nudt.edu.cn;
Contributing authors: wangzhichao@nudt.edu.cn; yanjunjun@nudt.edu.cn; liujie@nudt.edu.cn;

Abstract

CFDでは、メッシュの平滑化手法は、高精度な数値シミュレーションを実現するために、メッシュの品質を改良するために一般的に利用されている。具体的には、最適化ベースの平滑化は高品質なメッシュ平滑化に使用されるが、計算オーバーヘッドが大きい。先駆的な研究は、高品質なメッシュから平滑化手法を学習するために教師あり学習を採用することで、平滑化効率を向上させる。しかし、メッシュノードの次数を変化させた場合、メッシュノードの平滑化が困難であり、ノード入力シーケンスの問題に対処するためにデータ補強が必要である。さらに、ラベル付けされた高品質なメッシュが必要なため、提案手法の適用性がさらに制限される。本論文では、インテリジェントメッシュスムージングのための軽量ニューラルネットワークモデルであるGMSNetを紹介する。GMSNetは、ノードの近傍の特徴を抽出し、最適なノード位置を出力するために、グラフニューラルネットワークを採用する。平滑化の際にも、GMSNetが負の体積要素を生成するのを防ぐために、フォールトトレランス機構を導入する。軽量なモデルにより、GMSNetは程度の差こそあれメッシュノードを効果的に平滑化することができ、入力データの順序に影響されない。また、高品質なメッシュを必要としない新しい損失関数MetricLossを開発し、学習中に安定かつ迅速な収束を実現した。GMSNetを、一般的に用いられているメッシュスムージング手法と2次元三角メッシュ上で比較する。実験結果より、GMSNetは前モデルの5%のモデルパラメータで優れたメッシュ平滑化性能を達成し、最適化ベースの平滑化よりも8.62倍高速に達成することがわかった。

キーワード Unstructured Mesh, メッシュスムージング, グラフニューラルネットワーク, 最適化ベース Smoothing

1 Introduction

コンピュータ技術の急速な進歩に伴い、計算流体力学(CFD)は流体力学の原理を研究するための重要な手法として浮上してきた。その幅広い応用範囲は、航空宇宙、水工学、自動車工学、生物医学など、多様な分野に及んでいる[1-4]。通常、CFDシミュレーションは、支配的な物理方程式を離散化し、その後、離散化された方程式の大規模な代数系を解いて流体変数を求めることによって行われる。離散化はCFDの重要なステップであり、支配的な物理方程式の離散化と計算領域の離散化という2つの重要な側面を包含している[5]。後者のプロセスはメッシュ生成と呼ばれ、CFDにおいて基本的な役割を果たす。計算領域を、2次元領域のポリゴンや3次元領域のポリヘドラのような、重ならないメッシュ要素に分割することを含む[6]。生成されたメッシュの品質は、数値シミュレーションの収束性、精度、効率に大きな影響を与える。メッシュ要素の直交性、滑らかさ、分配性、密度分布は、解行列の安定性と収束性に大きく影響します[7]。その結果、高品質なメッシュ生成の探求は、CFD研究において活気に満ちた活発な分野であり続けている。実用的なメッシュ生成プロセスにおいて、最初に生成されたメッシュは、しばしばシミュレーションの要件を満たさない。メッシュの品質を向上させるために、メッシュスムージング、フェイススワップ、エッジスワップ、ポイント挿入/削除などのメッシュ品質向上技術が一般的に採用されています[8- 10]。このような手法の中で、メッシュの品質を向上させるために最もよく使われる手法はメッシュスムージング法である。

メッシュスムージングは、ヒューリスティックスムージングと最適化ベーススムージングの2種類に大別される[11]。代表的なヒューリスティック手法として、ラプラシアンスムージング[12]がある(図1a)。ラプラシアンスムージングでは、メッシュノードをStarPolygon(図1に示すように、ノードを含む多面体)のノードの座標の算術平均に配置して平滑化する。この方法は効率的であるが、StarPolygonが非凸の場合、負の体積要素を生成する可能性がある。

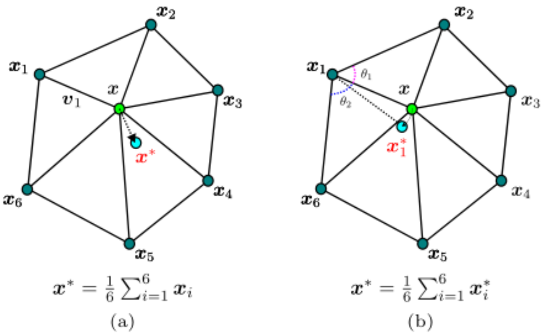


図1: a）ラプラシアン平滑化。点xのスター多角形はS(x) = {x_1, x_2,.. x_6 }を六角形とする。最適化されたメッシュノードは、StarPolygonのノードの座標の算術平均に位置する。b）角度ベースの平滑化。図はx_1に基づく最適位置を示している。これは、xをx_1の位置にある角度の角度二等分線に回転させることで、x_1^*が得られる。この処理をメッシュ内の他の点についても繰り返し、算術平均化により最終的に最適化された点を得る。

角度ベースの平滑化は、StarPolygonのノードの角度二等分線上にメッシュノードを配置することでメッシュ平滑化を実現します(図1bに示す)[13]。ノードベースの手法に加えて、Centroidal Voronoi tessellation (CVT) smoothing [14]は、Lloyd反復[15]によって各ノードのボロノイ領域を再計算し、平滑化のためにボロノイ領域の重心に点を再配置する。Lloydアルゴリズムの効率を高めるために、新しい点位置を計算する決定論的手法が提案されている[16]。ヒューリスティックに基づくメッシュ平滑化手法はシンプルで効率的であるが、その最適化能力は限られている。これらは逆要素につながる可能性があり、平滑化効果はヒューリスティック関数の設計に大きく依存する。一方、最適化ベースの手法は、局所領域におけるメッシュ品質評価指標を最適化することで、メッシュスムージングを実現する[17-20]。Parthasarathy and Kodiyalam [17]は、メッシュ平滑化問題を制約付き最適化問題として定式化し、メッシュノード位置を最適化するために反復最適化アルゴリズムを利用している。

その後の研究で異なるメッシュ品質評価指標と最適化手法が採用されているにもかかわらず、最適化ベースの手法は通常、メッシュスムージングのために最適化問題を反復的に解く必要があり、その結果、効率が低くなる。

近年、人工知能(AI)手法もメッシュ関連分野で広く利用されている。これらの研究成果の多くは、メッシュ品質評価[21-23]、メッシュ密度制御[24, 25]、メッシュ生成[26-28]、メッシュ精密化[29, 30]、メッシュアダパテーション[31-33]へのAI手法の適用に費やされている。しかし、AIベースのメッシュスムージングに関する研究は比較的少ない。Guoら[11]は、フィードフォワードニューラルネットワークを用いて最適化ベースの平滑化手法を模倣する教師あり学習アプローチを最初に導入した。提案モデルであるNN-Smoothingは、最適なノード位置を直接与えることで、最適化ベースのスモージングの効率を向上させる。しかし、フィードフォワードニューラルネットワークは固定次元の入力に悩まされ、異なる次数のメッシュノードに対しては別々のモデルが必要となり、異なる入力ノードのシーケンスに対してはデータ増強が必要となり、それによってモデルの学習コストが増加する。さらに、教師あり学習でモデルを学習させるために、高品質なメッシュ生成は、負担の大きい計算オーバーヘッドも発生させる。

このような制限を克服するために、本論文ではグラフニューラルネットワーク(GNN)[34]に基づく新しいメッシュスムージングモデルGMSNetを提示する。メッシュ平滑化の過程を学習するための軽量で効率的なGNNモデルを提案する。GMSNetは、メッシュノードの隣接ノードの特徴を抽出することで、最適化問題を解く際のオーバーヘッドを回避し、より良いノード位置を直接出力する。グラフニューラルネットワークが非構造化データを扱う能力により、GMSNetは単一のモデルで様々な程度のノードを平滑化でき、データ増強なしでノード入力シーケンス問題をエレガントに解決できることを示す。一度学習すれば、異なる形状の滑らかなメッシュに適用することができる。また、メッシュを平滑化する際に負の体積要素を導入しないために、シフト切り捨て操作を提案する。提案するGMSNetにとどまらず、メッシュ品質メトリクスに基づく新しい損失関数MeticLossを導入し、モデルを学習することで、高品質なメッシュを生成するオーバーヘッドをさらに排除する。GMSNetと一般的に使用されるメッシュ平滑化アルゴリズムについて、2次元三角メッシュ上で広範な実験を行う。実験結果は、我々のモデルが最適化ベースの平滑化と比較して8.62倍の高速化を達成し、同時に同程度の性能を達成し、他の全てのヒューリスティック平滑化アルゴリズムを凌駕することを示している。また、GMSNetは学習時に未見であったメッシュにも適用できることが示された。一方、従来のNN-Smoothingモデルと比較すると、GMSNetはモデルパラメータが5%しかないが、メッシュスムージング性能は優れている。また、提案するMetriLossとシフト切り捨ての有効性を比較実験により検証する。我々の貢献をまとめると以下のようになる:

1. インテリジェントメッシュスムージングのための軽量グラフニューラルネットワークモデルGMSNetを提案する。GMSNetは程度の差こそあれノードを平滑化することができ、データ入力順序の影響を受けない。さらに、GMSNetが負の体積要素を生成するのを防ぐために、フォールトトレランス機構であるシフトトランケーションを提供する。
2. メッシュ品質メトリクスに基づき、高品質なメッシュを必要としないモデルを学習するための新しい損失関数MetricLossを導入する。MetricLossはモデル学習中に安定した収束を示す。
3. 2次元三角メッシュを用いた広範な実験により、GMSNetの有効性を検証する。実験結果は、GMSNetが優れたメッシュスムージング性能を達成し、平均8.62倍のスピードアップで最適化ベースのスムージングを大幅に上回ることを示している。また、提案するMetricLossとシフト切り捨て操作の有効性を示すために、比較実験を行った。

本稿の残りの部分は以下のように構成されている。セクション2では、一般的に使用されているメッシュ平滑化手法と、メッシュ分野におけるニューラルネットワークの応用を紹介する。セクション3では、提案モデルを紹介し、メッシュデータの前処理、モデルのアーキテクチャ設計、損失関数、学習方法について詳細に説明する。セクション4では、提案モデルの性能をベースラインモデルと比較する実験を行い、損失関数とシフト切り捨ての有効性について議論する。

最後に、結論のセクションでは、論文全体を要約し、将来的に探求できる可能性のある研究を提案する。

2 リアルワーク

2.1 ヒューリスティックメッシュ平滑化

ラプラシアンスムージングは、ヒューリスティックメッシュスムージング法として最もよく使われる方法である。StarPolygonのノードの算術平均にノード座標を更新する。重み付きラプラシアンスムージング[35]は、スムージングプロセス中に隣接するノードやエッジに重みや重要度係数を追加導入し、スムージング効果をより制御し、メッシュの特定の特徴を保持する。スマートラプラシアンスムージング[18]は、各ノードの移動の前にチェックを行い、操作によってメッシュ品質が向上するかどうかを評価する。メッシュ品質が向上しない場合、メッシュノードの動きがスキップされ、より効率的な処理となる。角度ベースのメッシュ平滑化は、メッシュノードの角度を考慮することで、滑らかさを実現する。この方法では、ノードを回転させ、StarPolygonの各ノードの角度二等分線に合わせる。スターポリゴンの様々なノードの角度二等分線は一致しない可能性があるため、最終的なノード位置は追加計算が必要である。これは、ノード座標の平均を計算するか、最小二乗問題を解くことで実現できる。CVTスムージングは、メッシュノードによって定義されたボロノイ領域のセントロイドにノードを配置する。解法プロセスにおけるLloydアルゴリズムの効率を高めるために、以下に述べるように、セントロイドを計算するより効率的な方法が設計されている：

$$\boldsymbol{x}_i^* = \frac{1}{|\Omega_i|} \sum_{T_j \in \Omega_i} |T_j| \, C_j \tag{1}$$

ここで、x_i^*は新しいノードの位置を表し、$|\Omega_i|$はノードのスターポリゴンの総面積、$|T_j|$はj番目の三角形の面積、C_jはj番目の三角形の円周率である。

ヒューリスティック・スムージングと最適化ベースのスムージングの間に絶対的な境界がないことは注目に値する。別の観点からは、ヒューリスティックメッシュスムージングも最適化ベースのアプローチと見なすことができる。例えば、ラプラシアンスムージングは、エネルギー関数$E = \sum_{i=1}^{K} |v_i|^2$を最小化すると見なすことができる。ここで、$v_i$はxから$x_i$への辺を表し、KはStarPolygonのノード数である(図1a)。同様に、角度に基づくメッシュ平滑化は、エネルギー関数$E = \sum_{i=1}^{2K} \theta_i^2$を最小化すると見なすことができる。ここで、$\theta_i$はxから$x_i$までの辺と多角形の辺との間の角度である(図1bに示す)。ヒューリスティックメッシュスムージングの主な利点は、その効率にある。しかし、その平滑化性能は最適化ベースの平滑化性能に劣ることが多い。さらに、ヒューリスティックメッシュスムージングの有効性は、ヒューリスティック関数の設計に大きく依存する。

2.2 最適化に基づく平滑化

最適化に基づくメッシュ平滑化手法は、以下の問題として定式化できる：ノード位置x_iと制約条件の集合と目的関数fを持つ初期メッシュが与えられたとき、目的関数fを最小化する新しいノード位置x_i^*を見つけることが目的である。数学的には、これは次のように表すことができる：

$$\begin{aligned} &\boldsymbol{x}_i^* = \underset{\boldsymbol{x}_i}{\arg\min} f\left(\boldsymbol{x}_i, \mathbf{S}\left(\boldsymbol{x}_i\right)\right) \\ &\text{s.t.} \quad \boldsymbol{x}_i \in \mathcal{X} \end{aligned} \tag{2}$$

エネルギー関数$E = \sum_{i=1}^{K} |v_i|^2$を最小化すると見なすことができる、 f はメッシュ品質評価関数、X は x_i^* の制約を満たす実行可能集合、$S(x_i)$ は x_i のスターポリゴンのノード集合である。この関数の入力にはノード間の接続性も含まれていることを述べておくが、ここではわかりやすくするために省略する。評価関数の選択が異なると、メッシュの品質が異なることを反映して、メッシュの平滑化方法が異なる。一般的に使用される評価関数には、最大最小角度[36]、リスペクト比、歪み比などがある[17]。関数fが微分可能であれば、勾配$\Delta f = 0$とすることで、この制約付き最適化問題の最適点を解いて、明示的な式を得ることができる。しかし、ほとんどの場合、ラプラシアン平滑化や角度ベース平滑化のようにx_i^*の明示的な式を導くことは難しい。x_i^* の解法には反復法がよく使われるが、これは効率が悪い。したがって、最適なポジションを効率的に解く方法を開発することは、取り組むべき問題である。

2.3 メッシュ関連分野のニューラルネットワーク

ヒトの脳のニューロンからヒントを得て、人工ニューラルネットワークを利用して複雑な機能マッピングを学習する。彼らは、画像認識、音声認識、自然言語処理などのタスクに取り組み、機械学習や人工知能における広範なアプリケーションを発見している[37-40]。近年、ニューラルネットワークはメッシュに関連する様々な領域で重要な応用が見出されている。メッシュ品質評価の領域では、構造化メッシュ品質の自動評価を容易にするために、NACA-Marketデータセットとともに、畳み込みニューラルネットワークモデルであるGridNetがChenら[21]によって導入された。この概念を非構造化メッシュに拡張し、Wangら[22]はメッシュ品質評価にグラフニューラルネットワークを採用した。メッシュ分布の精密化を追求するために、Zhangら[24]は、従来のメッシュ生成ソフトウェアを強化するために人工ニューラルネットワークを採用し、領域全体の局所メッシュ密度の予測を可能にした。このアプローチは、広範なテストによって説得力のある実証がなされたように、四面体メッシュにさらに拡張された[25]。インテリジェントメッシュ生成の領域では、Daroyaら[26]が、点群からのグローバルな構造情報を活用して、高品質なメッシュ再構成を実現するアルゴリズムを発表した。同様に、Papageannopoulosら[27]は、メッシュ化された輪郭から抽出されたデータを利用してニューラルネットワークを学習し、メッシュ領域内のノードの数、配置、相互接続性を正確に近似することを可能にした。Chenら[28]は、新しい差分アプローチを用いて、構造化メッシュを生成するための教師なしニューラルネットワーク手法であるMGNetを導入し、有望な結果を得た。人工知能技術は、生成だけでなく、メッシュの改良や適応にも大きく貢献している。Bohn and Feischl [29]は、最適なメッシュ精密化アルゴリズムを学習するためにリカレントニューラルネットワークを採用し、広範な偏微分方程式を強化するための効果的なブラックボックスツールとしての能力を確立した。変分メッシュ適応を強化し、Tingfanら[31]は、更新されたメッシュ上の流れ場推定を迅速化するために、機械学習回帰モデルをシームレスに統合した。一方、Wallworkら[32]は、学習されたニューラルネットワークに支えられた、データ駆動型の目標指向メッシュ適応戦略を考案し、適応プロセスにおける計算コストのかかる誤差推定フェーズに効果的に取って代わる。さらに、FidkowskiとChen[33]は、計算メッシュの最適な異方性を確認するために、独創的に人工ニューラルネットワークを採用し、従来の方法と比較してメッシュ効率を向上させた。メッシュスムージングに関しては、NN-Smoothing [11]がフィードフォワードニューラルネットワークを用いた最適化ベースのメッシュスムージングを模倣し、最適化ベースのメッシュスムージングの効率を大幅に向上させている。しかし、別々のモデルの学習と高価な高品質メッシュ生成には、かなりの計算オーバーヘッドが発生する。

従来のディープラーニングアルゴリズムに加え、非構造化データを扱うための人工知能手法の学習能力を向上させるために、グラフニューラルネットワーク[34]が導入された。GNNはグラフ畳み込み[41]を利用して、グラフ上の特徴学習プロセスにトポロジカルな接続を組み込んでいる。メッシュはグラフデータとして自然に表現できるため、GNNはメッシュの精密化、流れ場のシミュレーション、乱流モデリングなど、様々な計算流体力学分野で広範な応用が見出されている[42-46]。したがって、メッシュスムージングにGNNを適用することは、有望かつ効果的な解決策であると主張する。

3 Methodology

3.1 問題の定式化

メッシュスムージング問題は、ノード間の接続性を維持しながら、そのノードの位置を調整することで、メッシュの品質を向上させることを含む。メッシュスムージング処理は、メッシュノードとその接続を入力とし、メッシュノードに新しい座標を出力として与え、メッシュ上で動作する関数として定義される。各メッシュノードについて、平滑化関数の入力は、それ自身と、そのノードの1リング近傍を構成するそのStarPolygonである。本論文では、メッシュをノードグラフとして定義する。具体的には、ノードx_0とそのスターポリゴンが与えられたとき、それをグラフ$G = (V, E)$で表現する。

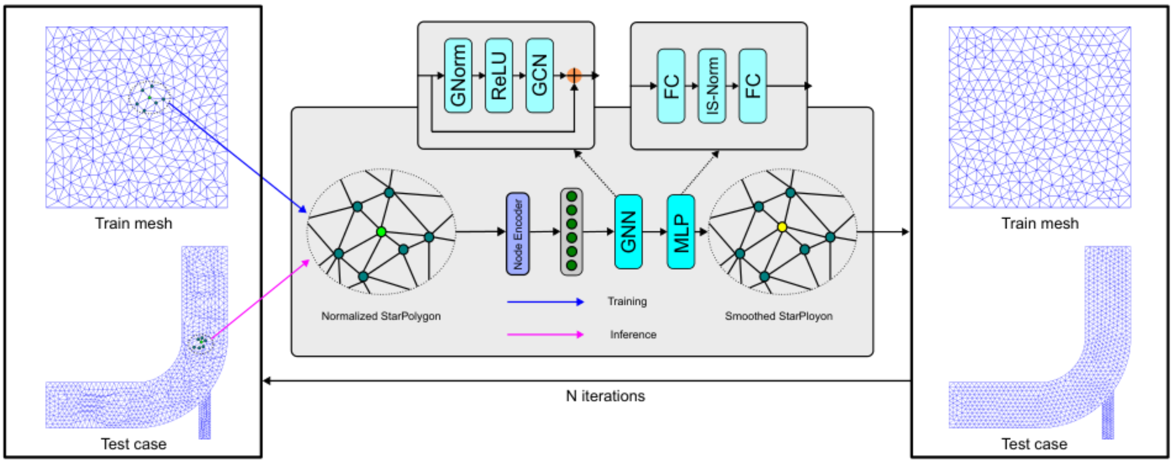


図2:GMSNetのアーキテクチャGMSNetは、各メッシュノードに対して最適化された位置を計算することにより、メッシュを平滑化する。このプロセスは、入力されたStarPolygonの正規化から始まる。次に、正規化された特徴量に対して特徴量変換を行い、StarPolygonノードからの情報をグラフ畳み込みにより統合する。最後に、モデルは完全連結(FC)層を通して最適化されたノード位置を予測する。図中、GNormはGraphNorm演算を表し、IS-NormはInstanceNorm演算を表す。モデルはメッシュ全体に対してN回反復し、平滑化操作を実行する。学習後、GMSNetはパイプのメッシュのような、以前に見たことのないメッシュに適用することができる。

x_n }はスターポリゴンのノードとノードを表し、x_iはスターポリゴンのノードiの座標、nはスターポリゴンのノード数、E = {(i, j) ¦ if x_i is connected with x_j }はノード間の接続を表す。典型的な平滑化処理は反復処理であり、t回目の反復ステップで、初期ノードグラフをG_t とし、中心ノードの最適化されたノード位置は次のように表すことができる:

$$\boldsymbol{x}_0^{t+1} = \mathcal{F}(\mathcal{G}_t) = \mathcal{F}(\mathcal{V}_t, \mathcal{E}) \tag{3}$$

本論文では、メッシュ平滑化のための関数Fを学習するために、グラフニューラルネットワークを採用する。

3.2 GMSNet

3.2.1 軽量モデルの設計

グラフニューラルネットワークに基づくメッシュスムージングモデルを図2に示す。メッシュノードとエッジからなるグラフが与えられたとき、我々の目標は、メッシュを平滑化するために、各メッシュノードに対してより最適なメッシュノード位置を計算することである。モデルでは、座標位置をノード特徴量として用い、X∈$_R^{(n+1)\times 2}$とする(StarPolygonのn個のノードと移動するための1個の空きノード)。

ノード間の接続性は隣接行列A∈_{R}^{(n+1)×(n+1)}で表される(ノードインデックスiはわかりやすくするために省略)。

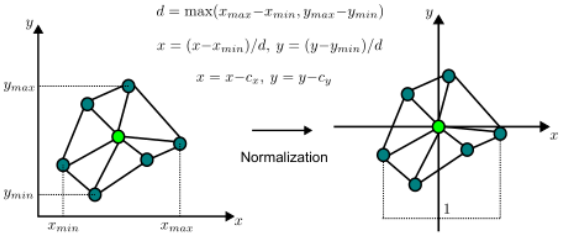


図3:データの正規化。処理するStarPolygonは座標軸の中心に対して正規化されている。図中、dはStarPolygonの最大サイズ、xとyはスケーリングする座標、c_xとc_yはスケーリング後のターゲットノードの正規化座標を表す。

モデルのスケール不変性を確保するため、処理中にノード正規化を適用する。このモデルは、Xに対してmin-max正規化を用いてノード入力を正規化し、0と1の範囲に制限し、その後、座標原点を中心にノードをセンタリングする変換を実行する。

モデル処理後、アフィン変換を採用し、ノードの位置を元のスケールに戻す。データの正規化を図3に示す。データを正規化した後、特徴量を線形層で変換し、次のように表すことができる。

$$X_h = \mathrm{Norm}(X)\mathbf{W}_l + \mathbf{b}_l \tag{4}$$

ここで、Normは正規化演算を表し、$W_l \in_R^{2\times H}$、b_lは線形層の変換行列とバイアス、Hは隠れ特徴の次元、$X_h \in_R^{(n+1)\times H}$は線形層の出力である。その後、残差グラフ畳み込みネットワーク(GCN)層[41, 47]を採用し、StarPolygonのノードの特徴に基づいて隠れ層の特徴を計算する。このプロセスでは、GraphNorm [48]を使用して線形層が出力する特徴を正規化し、その後、活性化層、畳み込み層、およびノードの隠れた特徴を計算するための和層が続く。この過程は次のように表すことができる:

$$\hat{X}_h = \mathrm{GraphNorm}(X_h) \tag{5}$$

$$X_g = \mathrm{GCN}(\mathrm{ReLU}(\hat{X}_h), \tilde{A}) + \hat{X}_h \tag{6}$$

$$= \tilde{A}[\mathrm{ReLU}(\hat{X}_h)]\mathbf{W}_g + \hat{X}_h \tag{7}$$

ここで、Ãは正規化隣接行列[1]、ReLUは活性化関数、W_gはGCN層のパラメータ、X_gは残差GCN層が出力する最終特徴量を表す。中心ノードの最終位置は、InstanceNorm [49]を用いた2層の完全連結ニューラルネットワークによって得られる。中心ノードのインデックスをi_cとすると、最適化されたノード位置は次式で与えられる:

$$\boldsymbol{x}^* = \mathrm{MLP}(X_g)[i_c] \tag{8}$$

平滑化アルゴリズムは、程度の差こそあれノードを処理する能力を持ち、ノード入力の順序に影響されないようにする必要がある。NN-Smoothingモデルでは、異なる次数のノードを別々のモデルで学習することで処理し、ノード入力の順序は、StarPolygonのリング内の開始ノードを変化させることでデータ増強することで対処する。

[1] $\hat{A} = A + I, \hat{D}$ is the degree matrix of $\hat{A}$, and $\tilde{A} = \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}$

しかし、グラフニューラルネットワークモデルの順列不変性により、入力順序(和関数や平均関数など)の順列にもかかわらず出力が変化しないため、GMSNetは別々のモデルを学習したり、データ補強を行うことなく、程度の異なるノードを効果的に扱うことができる。

3.2.2 モデル学習

NN-Smoothing法では、教師あり学習でモデルを学習し、最適化ベースの平滑化でラベル付き高品質メッシュを生成するが、これは時間のかかる作業である。一方、提案するGMSNetは、メッシュ平滑化のための最適化処理を直接学習する。このプロセスは、最適化ベースのスムージングと比較することで説明できる。最適化ベースの平滑化では、勾配降下を最適化手法とし、最適化されるメッシュ品質関数fが与えられると、その最小値を求めることが目的である。k回目の反復において、最適化アルゴリズムは位置を$x_{k+1} = x_k - \alpha_x$ f（x_k，S(x_0)）として更新する。ここで、$\triangle_x$ f（x_k，S(x_0)）はfの勾配、αはステップサイズである。x_k を繰り返し更新することで、関数は局所最適または大域最適 x^* に収束する。一方、NNSmoothingはメッシュ平滑化の最適点を直接予測する、すなわち、$x^{\hat{}*}$ = NN(x_0，S(x_0)）となる。しかし、このアプローチでは、最適化アルゴリズムによって生成されたラベル付き高品質メッシュが必要であり、時間がかかる。一方、GMSNetモデルは、メッシュノードの平滑化処理を学習するために、ラベル付きデータを必要としない。代わりに、学習プロセスはメッシュ要素の品質評価指標によって駆動され、それは次のように表すことができる:

$$\mathbf{W}^* = \arg\min_{\mathbf{W}} \mathcal{L}(\hat{\boldsymbol{x}}^*, \mathbf{S}(\boldsymbol{x}_0)) \tag{9}$$

$$= \arg\min_W L(F_W(x_0, S(x_0)), S(x))$$

ここで、Lはメッシュ品質指標を用いて構築された損失関数、F_Wは提案モデルを通して学習され、Wはモデルのパラメータである。損失関数はメッシュ要素の評価指標に基づいており、モデルはこの関数を最小化することでメッシュノードの位置を最適化する。

表1:最適化ベースのスムージング、NN-Smoothing、GMSNetの比較

Method	Speed	Labeled high-quality mesh	ノードの度数を変化させる	Node input order
OptimSmoothing[1]	Slow	Not acquiring	Not affected	Not affected
NN-Smoothing	Fast	Acquiring	Training separate models	Data augmentation
GMSNet	Fast	Not acquiring	Not affected	Not affected

[1] 最適化ベースの平滑化

このアプローチとNN-Smoothingの主な違いは、学習データにある。提案手法では、ラベルに高品質なメッシュは提供されない。その代わりに、学習プロセスはメッシュ品質メトリック関数の最小化のみに依存する。最適化ベースのメッシュスムージング手法との主な相違点は、このアプローチが最適化問題を解く必要なく、メッシュノードの最適化された位置を直接提供することであり、その結果、メッシュスムージングの効率が大幅に改善される。前述の3つの手法の包括的な比較を表1に示す。

3.2.3 メトリックロス

メッシュ要素の品質を評価するために、最大角度、最小角度、ヤコビアン行列、アスペクト比など、いくつかのメトリクスが利用可能である。我々の場合、メッシュ品質を評価するためにアスペクトm 2 +n 2 +l 2 ratio q = √を採用し、4 3Sここでm、n、lは三角形の辺、Sは三角形の面積である。正三角形の場合、この値は1であるが、縮退した三角形の場合、+∞に近づく。しかし、この指標の範囲が大きすぎるため、特に形状の悪い要素では勾配爆発を引き起こす可能性がある。この問題に対処するため、メッシュ要素の評価指標として $1 - \frac{1}{q}$ を用いる。正三角形の場合、この値は0であり、縮退三角形の場合、1である。損失関数の正式な定義は以下の通りである:

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{x}, \mathbf{S}(\boldsymbol{x})) &= \frac{1}{|\mathbf{S}(\boldsymbol{x})|} \sum_{i=1}^{|\mathbf{S}(\boldsymbol{x})|} (1 - \frac{1}{q_i}) \\
&= \frac{1}{|\mathbf{S}(\boldsymbol{x})|} \sum_{i=1}^{|\mathbf{S}(\boldsymbol{x})|} (1 - \frac{4\sqrt{3}S_i}{m_i^2 + n_i^2 + l_i^2})
\end{aligned} \tag{10}
$$

ここで、¦S(x)¦はStarPolygonのノード数、m_i, n_i, l_{i}は三角形T_iの辺、q_iはT_iの品質である。セクション4.4で、我々が設計した損失関数の有効性を検証した。

3.2.4 シフトの切り捨て

メッシュ生成とメッシュ平滑化の過程では、生成されたメッシュが負の体積要素を回避することが重要である。しかし、メッシュスムージングアルゴリズムは、図4に描かれているように、負の体積要素の生成につながることがある。例えば、ラプラシアンスムージングの場合、非凸のスターポリゴンを扱うと、負の体積要素が発生することがある(図4aに示すように)。同様に、CVTスムージング中、細長いメッシュ要素のボロノイ重心を計算すると、図4bに示すように、スターポリゴン領域から遠く離れた方向にシフトすることがあります(縮退したメッシュ要素の場合、重心位置は無限距離になることさえあります)。さらに、最適化アルゴリズムでは、メッシュ要素のスケールが異なる場合に一様なステップサイズを使用すると、最適化プロセスが最適位置をオーバーシュートする可能性があるため、負の体積要素の生成につながる可能性もある。

負の体積要素は、ニューラルネットワークベースの平滑化アルゴリズムの学習と推論の段階でも発生する可能性がある。負の体積要素の生成は、セクション4.3で検証したように、学習プロセスを混乱させない。モデル学習により、負の体積要素の発生は徐々に減少する。しかし、ニューラルネットワークの不確実性により、稀ではあるが、推論段階で負の体積要素を導入することが可能である。したがって、負の体積要素の発生を防ぐための方法が必要である。最も単純なアプローチは、負の体積要素をもたらす変位をゼロに設定することである。

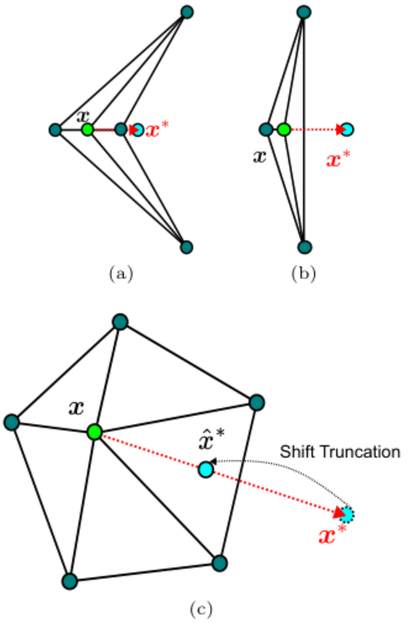


図4: a) 非凸のStarPolygonの場合、ラプラス平滑化によりStarPolygonの外側でノードを移動させることができる。b) CVTの平滑化には三角形の円周率の計算が必要である。高度に歪んだメッシュ要素の場合、その円周方向はStarPolygonから遠く離れており、負のボリューム要素になる。b) シフトの切り捨て。モデルの学習と推論の段階では、負の体積要素の生成を避けるためにシフトを切り捨てる。

しかし、このアプローチは、メッシュ平滑化の最適化目標である形状の悪いメッシュ要素の更新を妨げる。そこで、図4cに示すような負の体積要素を扱うために、線探索法を採用した。負の体積要素が生成されなくなるまで、負の体積要素を導入するシフトを半分に繰り返す。4.3節でシフト切り捨てがモデルに与える影響を調査する。

4 実験

4.1 実験セットアップ

本節では、提案するGMSNetと5つのベースライン平滑化手法(アルゴリズム)との包括的な性能比較を行った。評価したベースライン手法は以下の通りである: ラプラシアン平滑化[18]、角度ベースの平滑化[13]、CVT平滑化[14]、最適化ベースの平滑化、NNS平滑化[11]。以下は、各ベースラインモデルの実装の詳細である:

- すべてのモデルは非同期更新を採用しており、メッシュノードは各最適化ステップの後に直接更新されます(すべてのノードの更新を計算し、それらを一緒に更新するのとは対照的です)。
- 負の体積要素を防ぐために、スマートラプラシアンスムージングを採用した。
- 角度ベースの平滑化では、最終的なノード位置は、StarPolygonの各角度について計算された最適化された位置を平均化することによって得られた。
- CVTスムージングは、アルゴリズムの性能を向上させるために式1を採用した。
- 最適化ベースの平滑化では、3.2.3節で定義したMetricLossを目的関数として用い、最適化器としてAdam [50]を採用した。各ノードは最大20回の反復で最適化された。
- NN-Smoothingは原著論文の実装アプローチに従った。次数の異なるノードに対して、メッシュを平滑化するために異なるモデルを学習させた。さらに、3、4、5、6、7、8、9以外の次数のノードを扱うために、ラプラシアン平滑化を使用した。

合計20個のメッシュからなる2次元の三角形メッシュを用いてモデルを学習させた。データセットは、6:2:2の割合でトレーニングセット、検証セット、テストセットに分割された。各メッシュは、トレーニング前にランダムに生成された、異なるサイズと密度を持つ。

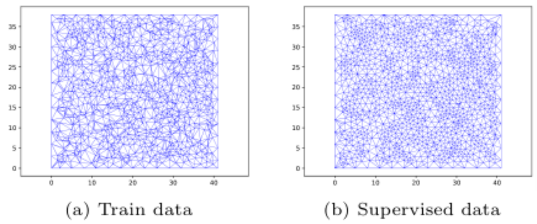
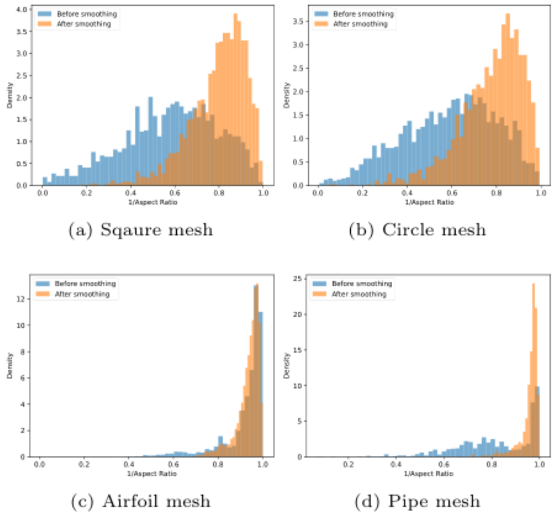


図5:NNSmoothingとGMSNetの学習用データセットのメッシュ例。GMSNetの学習時に教師ありデータは利用されない。

メッシュノードは幾何学的領域内にランダムに配置され、メッシュはドロネー三角形分割[51]を用いて生成される。メッシュの例を図5aに示す。

実験では、最適化ベースのスムージングから得られた最終的な最適化結果を、図5bに示すNNSmoothingモデルの学習ラベルとして利用した。同時に、同じデータセットを用いて、学習過程でラベルを組み込まずにGMSNetを学習させた。両ニューラルネットワークモデルとも、オプティマイザとして Adam [50] を採用し、初期学習率は 1e-2 とした。学習過程を通して、学習率は検証セットでの性能に基づいて動的に調整された。各トレーニングエポックにおいて、各メッシュから32個のメッシュノードをランダムにサンプリングし、モデルをトレーニングした。メッシュノードの部分的なサンプリングのみで学習したにもかかわらず、モデルは効果的に収束する。



(a) Sqaure mesh (b) Circle mesh

(c) Airfoil mesh (d) Pipe mesh

図6:GMSNetスムージング前後のメッシュ要素の品質分布。

4.2 実験結果

モデルの平滑化性能を評価するために、4つのメッシュケースでテストを行った。モデルの性能をより詳細にテストするために、図7の最初の列に示すように、高度に歪んだ要素を含むメッシュをテストケースとして構築した。最初の2つのメッシュでは、メッシュノードは幾何学的領域内で一様サンプリングによって生成された。後者の2つのメッシュは、まずメッシュ作成ソフトウェアを用いて高品質なメッシュを生成し、次に歪んだメッシュ要素を導入するために手動で調整した。

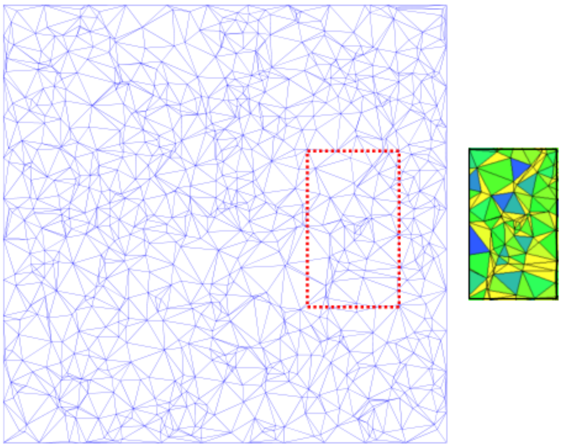
異なるモデル間の公平な比較を容易にするため、アルゴリズムを同じフレームワークで実装した。アルゴリズム間のバリエーションは、最適化された点位置の生成方法にある。本研究では、直列アルゴリズムを採用したことを述べておく。しかし、ラプラシアン平滑化のような単純なアルゴリズムでは、すべてのノードを同時に更新することは簡単で高速である。メッシュ品質の評価指標として、最小角度、最大角度、アスペクト比の逆数を選んだ。各ケースについて10回の実験を行い、1回の実験につき最大100回の平滑化反復を行った。に基づいて、最も平滑化されたメッシュを最終結果として選択した。

重み付き品質メトリック、これは次のように定義される:

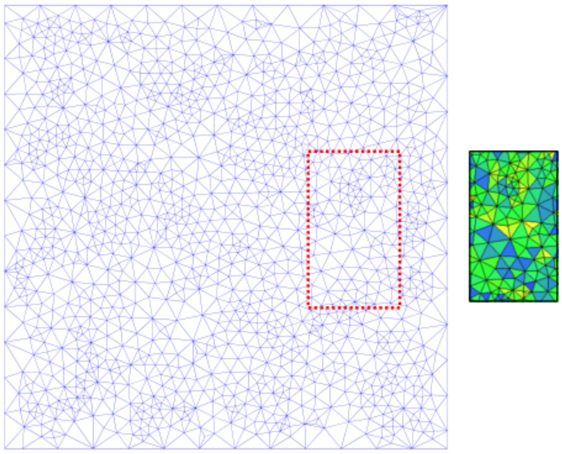
$$
\begin{aligned}
\hat{q} = \frac{1}{6}[&\frac{\alpha_{\text{mean}} + \alpha_{\text{min}} + 120 - \beta_{\text{max}} - \beta_{\text{mean}}}{60} \\
&+ \left(\frac{1}{q}\right)_{\text{mean}} + \left(\frac{1}{q}\right)_{\text{min}}]
\end{aligned} \tag{11}
$$

ここで、αは最小角度、βは最大角度、qはアスペクト比である。さらに、アルゴリズムの速度は、1つのメッシュノードを処理するのにかかる時間で測定される。GMSNetを用いたメッシュスムージングの結果を図7の2列目に示し、各アルゴリズムの性能の総合的な比較を表2にまとめた。

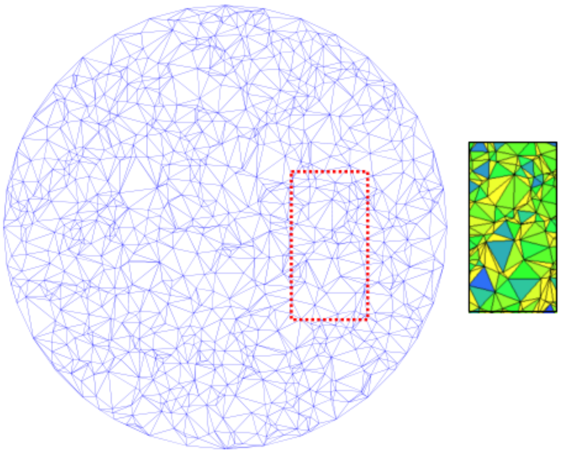
図7から、4つのテストケースすべてにおいて、我々の提案モデルがメッシュ要素の品質を大幅に改善していることがわかる。ノードの次数が適度に分布するメッシュの場合、図7fと7hに示すように、我々のアプローチは非常に滑らかなメッシュを生成する。さらに、我々のアルゴリズムは、図7bと7dに示すように、高度に歪んだメッシュに対する頑健性を保証する。表2に示すように、我々の提案するアルゴリズムは、ほとんどのヒューリスティックメッシュ平滑化手法を凌駕し、メッシュ要素の品質メトリクスは、最適化ベースのアルゴリズムを使用して得られた結果に近い。すべてのテストケースにおいて、我々のモデルは、以下の通り、NN-Smoothingモデルを概ね上回った。



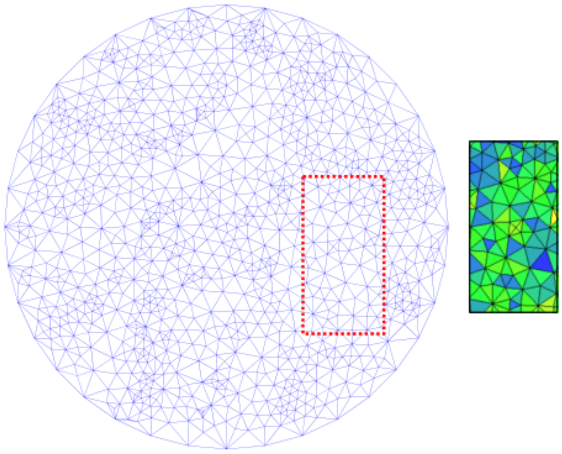
(a) 平滑化前の正方形メッシュ



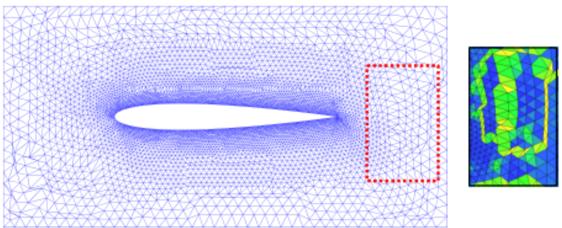
(b) 平滑化後の正方形メッシュ



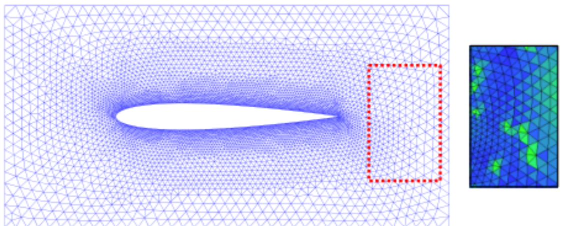
(c) 平滑化前の円メッシュ



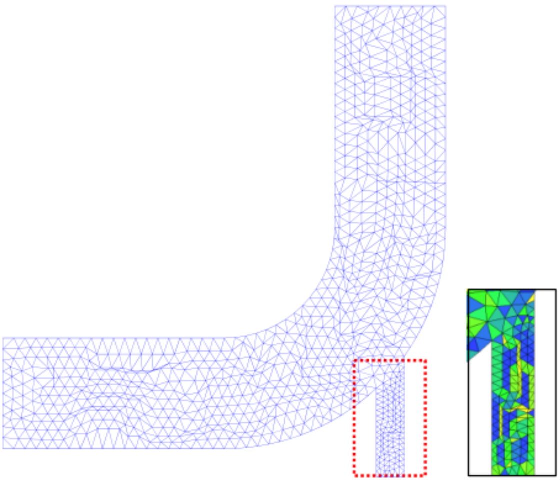
(d) 平滑化後の円メッシュ



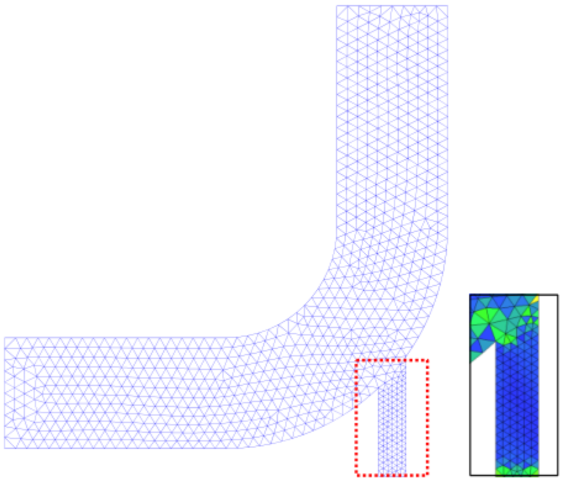
(e) Airfoil mesh before smoothing



(f) Airfoil mesh after smoothing



(g) Pipe mesh before smoothing



(h) Pipe mesh after smoothing

Fig. 7: Mesh smoothing results of GMSNet on the test cases. Mesh nodes in meshes a) and c) are randomly generated in the domain, and mesh nodes in meshes e) and g) are manually adjusted to introduce distorted elements. High-quality elements are colored blue, and low-quality elements are colored yellow.

表2:メッシュスムージングアルゴリズムの性能

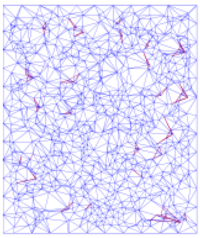
Mesh	Algorithm	Min. Angle		Max. Angle		$\frac{1}{\text{Aspect ratio}}$		s/per node
		min	mean	max	mean	min	mean	
Square	Origin	0.04	29.65	179.82	95.09	0.00	0.58	-
	LaplacianSmoothing	12.38	43.97	147.67	79.44	0.25	0.79	1.26E-03
	AngleSmoothing	10.52	41.26	148.6	81.76	0.22	0.76	2.36E-03
	CVTSmoothing	2.55	40.88	170.6	84.04	0.05	0.75	4.60E-03
	OptimSmoothing	16.33	43.94	136.99	80.7	0.32	0.79	2.29E-02
	NN-Smoothing	9.14	43.58	158.97	79.79	0.16	0.79	1.97E-03
	GMSNet	13.99	43.81	151.39	79.81	0.22	0.79	2.58E-03
Circle	Origin	0.29	30.38	179.05	94.60	0.01	0.59	-
	LaplacianSmoothing	1.43	42.86	174.53	81.19	0.03	0.78	1.21E-03
	AngleSmoothing	1.95	39.47	173.97	84.23	0.04	0.73	2.30E-03
	CVTSmoothing	3.19	39.98	172.98	85.59	0.05	0.73	4.35E-03
	OptimSmoothing	6.52	43.31	154.66	81.9	0.15	0.78	2.32E-02
	NN-Smoothing	1.26	41.1	176.04	83.86	0.03	0.75	1.88E-03
	GMSNet	2.2	43	169.21	81.29	0.05	0.78	2.38E-03
Airfoil	Origin	0.25	53.25	178.91	67.84	0.01	0.92	-
	LaplacianSmoothing	26.36	54.91	111.02	65.78	0.51	0.94	1.30E-03
	AngleSmoothing	20.26	53.49	118.76	66.64	0.42	0.93	2.54E-03
	CVTSmoothing	25.49	52.08	115.79	68.19	0.48	0.91	4.64E-03
	OptimSmoothing	30.53	54.9	108.85	65.72	0.54	0.94	1.66E-02
	NN-Smoothing	27.69	54.06	113.06	66.57	0.51	0.93	2.03E-03
	GMSNet	27.17	54.5	110.47	66.08	0.52	0.94	2.52E-03
Pipe	Origin	4.10	46.28	170.48	76.81	0.07	0.82	-
	LaplacianSmoothing	27.91	56.55	112.45	63.93	0.52	0.96	1.05E-03
	AngleSmoothing	24.28	54.62	101.93	65.69	0.53	0.94	1.93E-03
	CVTSmoothing	27.78	54.22	97.8	66.28	0.62	0.93	3.68E-03
	OptimSmoothing	32.39	56.41	106.07	64.14	0.57	0.96	1.79E-02
	NN-Smoothing	28.28	53.72	112.79	66.69	0.51	0.93	1.52E-03
	GMSNet	28.23	55.76	112.27	64.71	0.52	0.95	1.93E-03

NNSmoothingが7つのモデルを訓練する必要があるのに対して、我々は異なる度数のノードを平滑化するために1つのモデルだけを訓練したという事実。我々のモデルは、NN-Smoothingモデルと比較して、パラメータが5%しかない。効率性の観点から、提案手法は最適化ベースのアプローチと比較して、平均8.62倍の速度向上を示している。

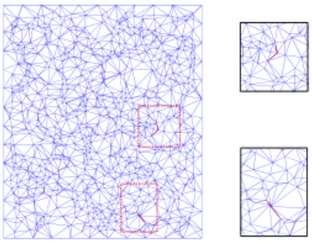
また、メッシュ要素の品質分布を調べることで、提案アルゴリズムの有効性を検証する。スムージング前後のメッシュ要素の品質分布を図6に示す。このアルゴリズムにより、高品質なメッシュ要素の割合が大幅に増加し、全体的なメッシュ品質が向上していることがわかる。以上より、メッシュ平滑化タスクに対する提案モデルの有効性と効率性が実験結果から実証された。

4.3 シフト切り捨ての本質

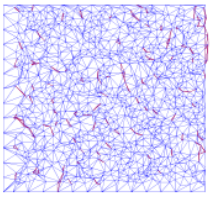
モデルにおけるシフト切り捨ての有効性を比較するために、学習の最初のエポック後の出力メッシュを可視化した。図8aに示すように、ノードの更新の大部分が負のボリューム要素をもたらしたことが観察される。しかし、モデルが学習を続けると、平滑化処理を学習し、負の体積要素の数が減少することが描かれている。



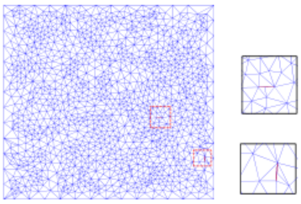
(a) Train mesh after model training for 1 epoch



(b) Train mesh after model training for 10 epochs



(c) Test mesh after model training for 1 epoch



(d) Test mesh after model training for 10 epochs

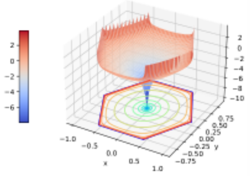
図8:シフト切り捨ての本質。図中の赤い要素は負の体積要素を表している。

を図 8b に示す。また、モデル予測におけるシフト切り捨ての影響も検討した。実験結果を図8cと図8cに示す。ある高度に歪んだ要素については、エポック学習したにもかかわらず、モデルが負の体積要素を生成する可能性があることがわかる。これは、予測段階でシフトの切り捨てが不可欠であることを示している。

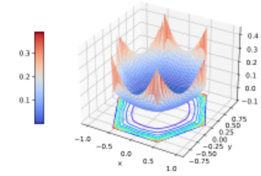
4.4 異なる損失関数の影響

3.2.3節では、モデル学習のための損失関数を設計した。本節では、MetricLossの有効性と適切な損失関数の選択方法について述べる。モデル学習時の損失関数として、一般的に使用されるメッシュ品質メトリクスの性能を比較する:

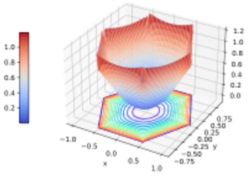
- 最小-最大角度損失
  $\mathcal{L}(\boldsymbol{x}, \mathbf{S}(\boldsymbol{x})) = \frac{1}{|\mathbf{S}(\boldsymbol{x})|} \sum_{i=1}^{|\mathbf{S}(\boldsymbol{x})|} \min\max(\theta_{ij})$, for $j = 1, 2, 3$, where θ_{ij} is the angle in triangle T_i.
- Aspect ratio loss:
  $\mathcal{L}(\boldsymbol{x}, \mathbf{S}(\boldsymbol{x})) = \frac{1}{|\mathbf{S}(\boldsymbol{x})|} \sum_{i=1}^{|\mathbf{S}(\boldsymbol{x})|} \frac{m_i^2+n_i^2+l_i^2}{4\sqrt{3}S_i}$, where the symbols are defined in Section 3.2.3.



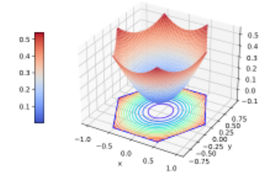
(a) アスペクト比損失



(b) 最小最大角度損失



(c) Cosine loss



(d) The proposed loss

図9:異なる損失関数のプロット。損失関数の範囲と滑らかさは、モデルの収束に影響を与える。図9aでは、z座標はlog(z)で与えられる。

- Cosine loss:
  $\mathcal{L}(\boldsymbol{x}, \mathbf{S}(\boldsymbol{x})) = \frac{1}{3|\mathbf{S}(\boldsymbol{x})|} \sum_{i=1}^{|\mathbf{S}(\boldsymbol{x})|} (\cos(\theta_{ij}) - \frac{1}{2})^2$
- MetricLoss:
  $\mathcal{L}(\boldsymbol{x}, \mathbf{S}(\boldsymbol{x})) = \frac{1}{|\mathbf{S}(\boldsymbol{x})|} \sum_{i=1}^{|\mathbf{S}(\boldsymbol{x})|} (1 - \frac{4\sqrt{3}S_i}{m_i^2+n_i^2+l_i^2})$

StarPolygonの中心ノードのバリエーションに関する前述の損失関数のプロットを図9に示す。我々が採用した損失関数は、入力が変化するにつれてより滑らかな変換を示し、最適化プロセスを容易にすることが観察できる。また、アスペクト比損失と最小-最大角度損失は、損失関数として採用するのに適していないことがわかる。非常に歪んだ要素に遭遇した場合、前者は数値が正の無限大に近づき、損失が急激に増加し、学習が不安定になる。一方、後者は、高度に歪んだ要素に直面すると損失が急激に変化するが、中央のノードが最適位置の近傍に位置する場合、損失関数は比較的平坦なままである。このような状況は、最適化プロセスの妨げにもなる。元の角度ベースの損失関数は学習が難しいが、図9cに示すように、余弦関数を用いて変換した損失関数はより安定している。同じモデル構成と学習構成を採用し、異なる損失関数を用いてモデルを学習させた。

表3:パイプメッシュとスクエアメッシュにおけるコサインロスと提案ロスの比較

		Min. Angle		Max. Angle		$\frac{1}{\text{Aspect ratio}}$	
Mesh	Algorithm	min	mean	max	mean	min	mean
Pipe	Cosine loss	26.77	55.71	110.43	64.67	0.54	0.95
	MetricLoss	26.37	55.56	107.73	64.82	0.56	0.95
Square	Cosine loss	11.04	43.76	146.87	79.61	0.25	0.79
	MetricLoss	13.53	44.06	150.46	79.65	0.22	0.79

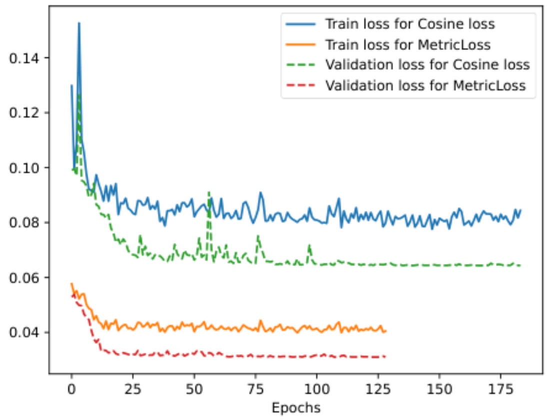


図10: 損失関数の違いによる比較。最小-最大角度とアスペクト比に基づく損失関数は、初期エポック中に学習失敗に遭遇した。一方、MetricLossと変換されたCosine損失関数は収束した結果を得た。

損失関数を変えたモデルの訓練損失と検証損失を図10に示す。MetricLossとcosine損失関数は、数回の繰り返しでモデルが急速に収束するため、モデルの学習に適していることがわかる。さらに、MetricLossは学習過程においても驚くほど安定している。異なる損失関数に基づく2つのモデル間の実験を表3に示す。両モデルは同様の性能を示していることがわかる。

他の2つの損失関数で学習する場合、モデルは収束しないか、激しい振動を示す。このことは、高度に歪んだ要素に遭遇したときに、より大きな数値範囲や数値の急激な変化を持つメッシュ品質メトリクスを採用することは、モデルの学習に不利であることを示している。要約すると、メッシュスムージングモデルの学習には様々な損失関数を用いることができる。

メッシュ品質メトリクスによって損失関数を構築する場合、歪んだ要素に遭遇したときの関数の挙動を考慮し、数値の範囲が大きくなったり、急激な変化があったりする損失関数が選択されないようにする必要がある。不適切なメッシュ品質メトリクスは、最小-最大角度損失と余弦損失によって示されるように、変換してモデル学習に採用することもできる。

5 Conclusion

本論文では、インテリジェントメッシュスムージングのためのグラフニューラルネットワークモデルGMSNetを提案する。提案モデルは、メッシュノードの近傍を入力とし、メッシュノードの平滑化された位置を直接出力するように学習することで、最適化ベースの平滑化に伴う計算オーバーヘッドを回避する。また、負の値要素を防ぐために、フォールトトレランス機構であるシフトトランケーションが適用されている。軽量設計により、GMSNetは程度の異なるメッシュノードに適用でき、データ入力順序の影響を受けない。また、メッシュ品質メトリクスに基づく新しい損失関数MetricLossを導入することで、モデルを学習するための高品質なメッシュ生成コストを不要にする。2次元三角メッシュを用いた実験により、我々の提案モデルは、最適化ベースの平滑化と比較して、平均8.62倍の高速化で優れた平滑化性能を達成することが実証された。また、GMSNetはNNSmoothingモデルよりも、わずか5%のモデルパラメータで優れた性能を達成することが示された。また、MetricLossが高速で安定したモデル学習を実現することを説明し、実験を通してシフト切り捨て操作の本質を示す。

今後の課題としては、提案手法を他の種類のメッシュ要素に適用し、

表面メッシュや体積メッシュに拡張することである。また、優れたメッシュ平滑化効果を得るために、メッシュ平滑化プロセスにエッジの反転やメッシュ密度の修正を導入することも、検討する価値のあるアプローチである。

補足情報である。

Acknowledgments. This research work was supported in part by the National Key Research and Development Program of China (2021YFB0300101).

Declarations

References

[1] Spalart, P.R., Venkatakrishnan, V.: On the role and challenges of cfd in the aerospace industry. The Aeronautical Journal **120**(1223), 209–232 (2016)

[2] Bridgeman, J., Jefferson, B., Parsons, S.A.: The development and application of cfd models for water treatment flocculators. Advances in Engineering Software **41**(1), 99–109 (2010)

[3] Damjanović, D., Kozak, D., Živić, M., Ivandić, Ž., Baškarić, T.: Cfd analysis of concept car in order to improve aerodynamics. Járműipari innováció **1**(2), 108–115 (2011)

[4] Samstag, R.W., Ducoste, J.J., Griborio, A., Nopens, I., Batstone, D., Wicks, J., Saunders, S., Wicklein, E., Kenny, G., Laurent, J.: Cfd for wastewater treatment: an overview. Water Science and Technology **74**(3), 549–563 (2016)

[5] Darwish, M., Moukalled, F.: The Finite Volume Method in Computational Fluid Dynamics: an Advanced Introduction with OpenFOAM® and Matlab®, pp. 110–128. Springer, Berlin (2016)

[6] Baker, T.J.: Mesh generation: Art or science? Progress in Aerospace Sciences **41**(1), 29–63 (2005)

[7] Knupp, P.M.: Algebraic mesh quality metrics. SIAM journal on scientific computing **23**(1), 193–218 (2001)

[8] Freitag, L.A., Ollivier-Gooch, C.: Tetrahedral mesh improvement using swapping and smoothing. International Journal for Numerical Methods in Engineering **40**(21), 3979–4002 (1997)

[9] Freitag, L.A., Ollivier-Gooch, C.: A comparison of tetrahedral mesh improvement techniques. Technical report, Argonne National Lab.(ANL), Argonne, IL (United States) (1996)

[10] Prasad, T.: A comparative study of mesh smoothing methods with flipping in 2d and 3d. PhD thesis, Rutgers University-Camden Graduate School (2018)

[11] Guo, Y., Wang, C., Ma, Z., Huang, X., Sun, K., Zhao, R.: A new mesh smoothing method based on a neural network. Computational Mechanics, 1–14 (2021)

[12] Herrmann, L.R.: Laplacian-isoparametric grid generation scheme. Journal of the Engineering Mechanics Division **102**(5), 749–756 (1976)

[13] Zhou, T., Shimada, K.: An angle-based approach to two-dimensional mesh smoothing. IMR **2000**, 373–384 (2000)

[14] Du, Q., Gunzburger, M.: Grid generation and optimization based on centroidal voronoi tessellations. Applied mathematics and computation **133**(2-3), 591–607 (2002)

[15] Lloyd, S.: Least squares quantization in pcm. IEEE transactions on information theory **28**(2), 129–137 (1982)

[16] Du, Q., Faber, V., Gunzburger, M.: Centroidal voronoi tessellations: Applications and algorithms. SIAM review **41**(4), 637–676 (1999)

[17] Parthasarathy, V., Kodiyalam, S.: A constrained optimization approach to finite element mesh smoothing. Finite Elements in

Analysis and Design **9**(4), 309–320 (1991)

[18] Field, D.A.: Laplacian smoothing and delaunay triangulations. Communications in applied numerical methods **4**(6), 709–712 (1988)

[19] Canann, S.A., Tristano, J.R., Staten, M.L.: 三角メッシュ、四辺形メッシュ、四辺形メッシュのためのラプラシアンと最適化に基づく平滑化の組み合わせへのアプローチ。IMR 1, 479-94 (1998). 出版社 Citeseer

[20] フィールド、D.A.: ラプラシアン平滑化とデラウネイ三角形分割。応用数値計算法における通信 4(6), 709-712 (1988)

[21] Chen, X., Liu, J., Pang, Y., Chen, J., Chi, L., Gong, C.: Developing a new mesh quality evaluation method based on convolutional neural network. Engineering Applications of Computational Fluid Mechanics **14**(1), 391–400 (2020)

[22] Wang, Z., Chen, X., Li, T., Gong, C., Pang, Y., Liu, J.: Evaluating mesh quality with graph neural networks. Engineering with Computers **38**(5), 4663–4673 (2022)

[23] Chen, X., Liu, J., Gong, C., Li, S., Pang, Y., Chen, B.: Mve-net: An automatic 3-d structured mesh validity evaluation framework using deep neural networks. Computer-Aided Design **141**, 103104 (2021)

[24] Zhang, Z., Wang, Y., Jimack, P.K., Wang, H.: Meshingnet: A new mesh generation method based on deep learning. In: International Conference on Computational Science, pp. 186–198 (2020). Springer

[25] Zhang, Z., Jimack, P.K., Wang, H.: Meshingnet3d: Efficient generation of adapted tetrahedral meshes for computational mechanics. Advances in Engineering Software **157**, 103021 (2021)

[26] Daroya, R., Atienza, R., Cajote, R.: Rein: Flexible mesh generation from point clouds. In: Proceedings of the IEEE/CVF Conference コンピュータビジョンとパターン認識ワークショップ, pp.352-353 (2020)

[27] Papagiannopoulos, A., Clausen, P., Avellan, F.: How to teach neural networks to mesh: Application on 2-d simplicial contours. Neural Networks **136**, 152–179 (2021)

[28] Chen, X., Li, T., Wan, Q., He, X., Gong, C., Pang, Y., Liu, J.: Mgnet: a novel differential mesh generation method based on unsupervised neural networks. Engineering with Computers **38**(5), 4409–4421 (2022)

[29] Bohn, J., Feischl, M.: Recurrent neural networks as optimal mesh refinement strategies. Computers & Mathematics with Applications **97**, 61–76 (2021)

[30] Paszyński,M.、Grzeszczuk,R.、Pardo,D.、Demkowicz,L.:ディープラーニング駆動の自己適応hp有限要素法。In: 計算科学国際会議, pp.114-121 (2021). シュプリンガー

[31] Tingfan, W., Xuejun, L., Wei, A., Huang, Z., Hongqiang, L.: A mesh optimization method using machine learning technique and variational mesh adaptation. Chinese Journal of Aeronautics **35**(3), 27–41 (2022)

[32] Wallwork, J.G., Lu, J., Zhang, M., Piggott, M.D.: E2n: error estimation networks for goal-oriented mesh adaptation. arXiv preprint arXiv:2207.11233 (2022)

[33] Fidkowski, K.J., Chen, G.: Metric-based, goal-oriented mesh adaptation using machine learning. Journal of Computational Physics **426**, 109957 (2021)

[34] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems **32**(1), 4–24 (2020)

[35] Vollmer, J., Mencl, R., Mueller, H.: Improved laplacian smoothing of noisy surface meshes. In: Computer Graphics Forum, vol. 18, pp. 131–138 (1999). Wiley Online Library

[36] Xu, K., Gao, X., Chen, G.: Hexahedral mesh quality improvement via edge-angle optimization. Computers & Graphics **70**, 17–27 (2018)

[37] Reddy, D.R.: Speech recognition by machine: A review. Proceedings of the IEEE **64**(4), 501–531 (1976)

[38] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. nature **521**(7553), 436–444 (2015)

[39] Pak, M., Kim, S.: A review of deep learning in image recognition. In: 2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT), pp. 1–3 (2017). IEEE

[40] Chowdhary，K.，Chowdhary，K.：自然言語処理.人工知能の基礎，603-649（2020）

[41] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

[42] Song, W., Zhang, M., Wallwork, J.G., Gao, J., Tian, Z., Sun, F., Piggott, M., Chen, J., Shi, Z., Chen, X., *et al.*: M2n: mesh movement networks for pde solvers. Advances in Neural Information Processing Systems **35**, 7199–7210 (2022)

[43] Lino, M., Cantwell, C., Bharath, A.A., Fotiadis, S.: Simulating continuum mechanics with multi-scale graph neural networks. arXiv preprint arXiv:2106.04900 (2021)

[44] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., Battaglia, P.W.: Learning mesh-based simulation with graph networks. arXiv preprint arXiv:2010.03409 (2020)

[45] Han, X., Gao, H., Pfaff, T., Wang, J.-X., Liu, L.-P.: Predicting physics in mesh-reduced space with temporal attention. arXiv preprint arXiv:2201.09113 (2022)

[46] Peng，W.，Yuan，Z.，Wang，J.：乱流シミュレーションのための注意強化型ニューラルネットワークモデル.流体の物理 34(2) (2022)

[47] Li, G., Muller, M., Thabet, A., Ghanem, B.: Deepgcns: Can gcns go as deep as cnns? In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9267–9276 (2019)

[48] 蔡、T.、羅、S.、徐、K.、何、D.、劉、T.-y.、王、L.：Graphnorm：グラフニューラルネットワークの学習を加速するための原理的なアプローチ。In：機械学習国際会議，pp.1204-1215（2021）．PMLR

[49] Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)

[50] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[51] Lee,D.-T.、Schachter,B.J.:脱則的三角測量を構築するための2つのアルゴリズム。国際コンピュータ情報科学ジャーナル 9(3)，219-242（1980）