Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

GDEA: Global-Weight Deep Equilibrium Attention for Finite Element Systems

Junghun Lee

junghunl@andrew.cmu.edu

Carnegie Mellon University

Conrad Tucker

Carnegie Mellon University

Article

Keywords: Surrogate model, Graph Neural Network, Finite element method

Posted Date: April 14th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9316051/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: There is **NO** Competing Interest.

GDEA: Global-Weight Deep Equilibrium Attention for Finite Element Systems

Junghun Lee[1*] and Conrad Tucker[1]

[1*]Mechanical Engineering, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, 15232, PA, USA.

*Corresponding author(s). E-mail(s): junghunl@andrew.cmu.edu;
Contributing authors: conradt@andrew.cmu.edu;

Abstract

Graph neural networks have been explored as surrogate models for the finite element method due to the computational cost. However, many models fail to effectively capture the influence of distant nodes. This is critical in surrogate models because boundary conditions at a distance can significantly influence the results. To mitigate these issues, we propose iterative message passing using a global-weight matrix assembled via the direct stiffness method. This method is implemented through a deep equilibrium model with Anderson acceleration to ensure fast convergence and low memory cost. Our model is evaluated on published FEM datasets as well as a new dataset, Deformed ABC, featuring diverse geometries, materials, and boundary conditions. Our model outperforms others across all criteria, including a 12% lower average RMSE and a 5% higher average R^2 on a complex new dataset, but differences are marginal due to saturation of the benchmark across all models in simpler datasets.

Keywords: Surrogate model, Graph Neural Network, Finite element method

1 Introduction

The Finite Element Method (FEM) is a fundamental computational tool for engineering design analysis and scientific modeling described by Partial Differential Equations (PDEs) [1]. Recently, its impact on engineering design has become so pervasive that FEM is now considered essential for many engineering design and analysis tasks [2, 3]. One of the major limitations of FEM is high computational cost and time [1, 4, 5].

Reducing the computational cost of simulations meets the industry's demand for fast, near-real-time simulations of engineering problems [1]. Among different modeling methods, deep neural networks have achieved significant success, as they can capture complex, high-dimensional behaviors directly from data, whereas other machine learning models often rely on feature engineering and struggle with such data [4, 5]. Additionally, the computational cost of deep neural networks increases more slowly with mesh resolution than that of FEM, rendering them efficient surrogate models [6]. Computational cost of neural network–based surrogate models scales linearly with the number of operations [7], whereas the cost of FEM scales quadratically with the number of elements [8, 9]. However, ensuring high predictive accuracy remains a critical challenge for FEM surrogate models. Inaccurate FEM predictions can substantially increase the time and resources required for design and may also lead to suboptimal designs or violations of design constraints [2, 10]. The previous case study demonstrates that reducing FEM error from 2.5% to 1% reduces the suboptimal area by 96% and stress constraint violations by 70% in a deterministic optimization based on FEM [10]. These findings highlight that even small prediction errors can significantly affect design outcomes, underscoring the need for accurate surrogate models.

Accurately capturing interactions between nodes is critical for achieving accurate neural network–based FEM surrogate models. Graph-based geometric deep learning methods, such as graph neural networks (GNNs) [11], leverage message passing to capture interactions among nodes. Their flexibility with respect to input data enables FEM solution prediction across different mesh sizes. However, their ability to capture long-range dependencies remains limited, as they may suffer from under-reaching and over-smoothing. First, under-reaching is a structural limitation in which information cannot propagate between topologically distant nodes when the number of GNN layers is less than the graph diameter [12]. This limitation is critical in FEM surrogate models, where capturing long-range topological dependencies is essential, as solutions can be strongly influenced by boundary conditions applied at distant nodes. In such cases, especially for high-resolution meshes or diverse boundary conditions, a fixed number of message-passing layers may be insufficient to propagate information across the entire graph. On the other hand, unnecessary message-passing can lead to inefficiency and instability. To address these issues, we iteratively propagate information along the edges using global-weight matrices until nodal features converge, ensuring efficient propagation to distant nodes. However, repeated message passing or deeper GNN layers may suffer from over-smoothing, in which node features become indistinguishable [13]. This can be a critical issue in FEM scenarios where nodal solutions can differ significantly due to complex geometry, inhomogeneous material, and boundary conditions. In our model, we assign different weights to neighboring nodes and to nodes themselves to mitigate this issue [14, 15]. These weights place greater emphasis on influential nodes and neglect others. Building on this idea, we further enhance the weighting strategy by constructing a global stiffness matrix using the direct stiffness method, one of the most popular methods for constructing a stiffness matrix in FEM [16]. This assembly process results in diagonal dominance in the global-weight matrix as the self-connection weight increases with the number and strength of connections. This creates a form of informational inertia, helping each node preserve its original

identity from the influence of neighboring features. Finally, removing MLP components at each message-passing step can prevent other issues observed in deep GNNs, such as rank collapse [17] and gradient vanishing [18], while maintaining a deep and flexible iteration.

Recently, augmenting graph connectivity with a reduced graph obtained via sampling or pooling has been proposed to capture the influence of distant nodes [19, 20]. However, leveraging these reduced graphs can introduce performance inconsistencies and distort geometries, leading to an incomplete representation of the original mesh topology. In contrast, our model accounts for mesh connectivity without any modifications, as in the numerical FEM method. Graph Transformers are another emerging approach for capturing long-range dependencies in graphs [21]. By leveraging global attention, they allow interactions between distant nodes regardless of graph distance. However, they typically rely on implicit mechanisms, such as positional or structural encoding [22], rather than directly considering mesh topology. In addition, their computational complexity often scales quadratically with the number of nodes, which can limit scalability [23, 24]. To reduce computational complexity, linear attention has been introduced to approximate attention in graph transformers, achieving linear complexity [24]. However, such approaches primarily model relationships among nodal features from a global perspective, which can limit their ability to capture interactions among nodes [23].

In this work, we reformulate message passing as an iterative update using a global-weight matrix to overcome the limitations of GNNs and graph transformers. In this method, a two-by-two local weight matrix is predicted for each edge, and these matrices are then assembled into a global-weight matrix via the direct stiffness method. The global-weight matrix is used to update the node features via message passing until the node features converge. This approach introduces two major challenges. First, accumulating gradients over iterations can easily lead to memory shortages. Accordingly, we employ a Deep Equilibrium model (DEQ) to maintain constant memory usage regardless of the number of iterations. The converged equilibrium point in DEQ effectively encodes the result of an unbounded number of forward passes [25]. Second, iterative updates to node features and the DEQ's backward solver may require excessive iterations, slowing the model. To mitigate this issue, DEQ is often combined with acceleration methods to improve convergence speed [25]. In this study, we adopt Anderson acceleration, which leverages prior records to accelerate convergence to the solution [26, 27].

Our model, Global-Weight Deep Equilibrium Attention (GDEA), is compared with state-of-the-art GNN-based and Graph Transformer-based FEM surrogate models, as these methods can handle meshes of varying sizes. The comparison is conducted across multiple FEM datasets, including the Deformed ABC dataset. We introduce Deformed ABC to address the limitations of other FEM datasets, which often lack diversity in geometry, materials, and boundary conditions [19, 22, 28–32]. Dataset diversity is essential for achieving uniform coverage of the design space and mitigating bias that may limit model generalization [33]. Models that generalize across diverse simulation scenarios can reduce the need to train separate models for each scenario [34]. We also benchmark our model against state-of-the-art models on two publicly available

datasets: the simulated additive manufacturing displacement fields dataset [19] and the simulated flag-waving-in-the-wind dataset [32] to evaluate its performance across datasets with different mechanical systems and mesh types. Furthermore, we evaluate our model through ablation studies to understand the contribution of each component.

2 Results

2.1 Deformed ABC

We generate Deformed ABC to address the limitations of other FEM datasets, which often lack diversity in geometry, materials, and boundary conditions. Geometries are sampled from the ABC dataset [35], which contains one million Computer-Aided Design (CAD) models for research in geometric deep learning, and material data and boundary conditions are randomly selected. Deformed ABC consists of 71,508 samples with unique geometries, random material properties, and boundary conditions, ensuring diversity across all factors. The average mesh volume is 216,260.75 mm^3, with a median of 17,875.02 mm^3. The average number of elements is 817, with a median of 345. Fig. 1 (g) demonstrates that larger mesh volumes correspond to more elements. The average FEM execution time is 7.81 seconds, with the median being 4.02 seconds. Fig. 1 (h) shows that FEM execution time increases rapidly with the number of elements. The average of the maximum displacement of the FEM solution is 0.04 mm, and the median is 0.02 mm. 1 (e) and (f) shows that Young's modulus and Poisson's ratio are sampled uniformly to ensure broad coverage of material properties, with Young's modulus sampled on a logarithmic scale. The average Young's modulus is 86.83 GPa, with a median of 3.13 GPa, and the average Poisson's ratio is 0.26, with a median of 0.26. Figure 3 demonstrates an example of FEM data.

2.2 GDEA: Global-Weight Deep Equilibrium Attention

Our model GDEA is compared against state-of-the-art GNN-based and Graph Transformer-based FEM surrogate models, as they can handle meshes of varying sizes, which is essential given that our datasets include meshes with varying resolutions. Table 1 summarizes the model's performance on the Deformed ABC. Our model achieves the best performance across all criteria, as shown in Table 1, compared with other state-of-the-art models. Compared to the next-best model GINO [22], our model improves the average RMSE by 12% and average R^2 by 5%. Additionally, the trends in the first quarter (Q1), Median, and the third quarter (Q3) show that the improvement is more pronounced in finite element systems, where prediction accuracy is generally lower. Compared to the next-best model, GINO [22], our model improves R^2 Q1 and RMSE Q3 by 8% and 16%, respectively, whereas the improvements in R^2 Q3 and RMSE Q1 are only 2% and 5%, respectively. The same trend is observed across comparisons with other models. Finally, our model exhibits the smallest standard deviation, improving the R^2 standard deviation by 4% and the RMSE standard deviation by 3% compared to the next best model TAG-Unet [19]. The average execution time per geometry of our model is 0.023 seconds, resulting in a 99.7% reduction compared to the numerical FEM execution time in the data generation process (7.81

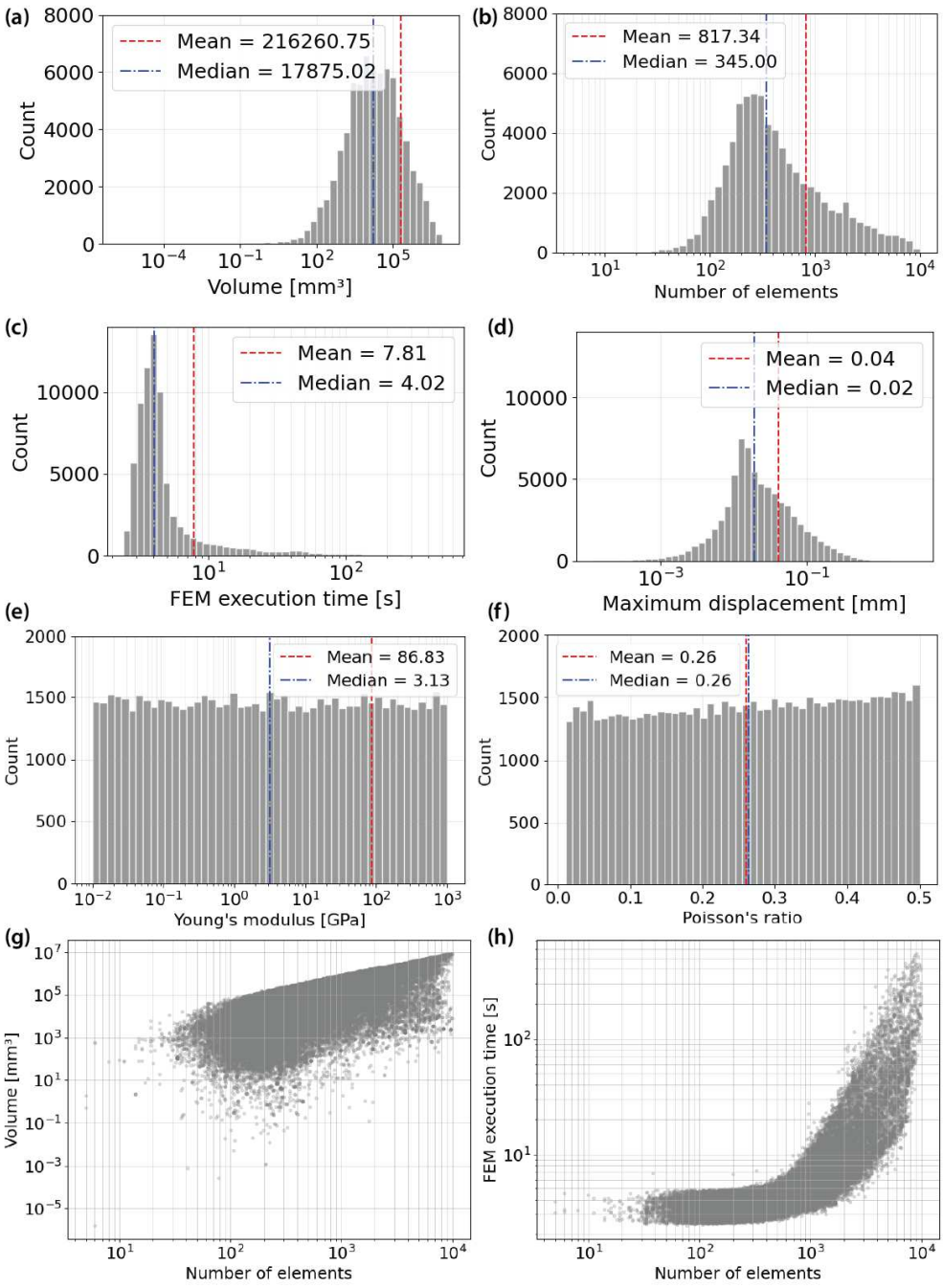


Fig. 1 This figure illustrates the distributions of (a) mesh volume, (b) number of mesh elements, (c) FEM execution time, (d) maximum displacement in the finite element system, (e) Young's modulus, and (f) Poisson's ratio. Panels (g) show the relationship between mesh volume and the number of elements, and (h) depict the relationship between the number of elements and FEM execution time.

seconds), as presented in section 2.1. Our model preserves the fast inference advantage of GNNs, achieving significantly faster runtime than numerical FEM.

Table 1 Comparison to the state-of-the-art models on the Deformed ABC dataset

Model	Year	R^2					RMSE				
		Mean (↑)	SD (↓)	Q1 (↑)	Median (↑)	Q3 (↑)	Mean (↓)	SD (↓)	Q1 (↓)	Median (↓)	Q3 (↓)
GINO [22]	2023	0.80	0.31	0.77	0.91	0.96	6.86	4.91	3.36	5.12	8.77
GNOT [36]	2023	0.77	0.37	0.74	0.91	0.96	7.15	5.41	3.34	5.23	9.18
Transolver [37]	2024	0.51	0.38	0.31	0.61	0.80	11.9	5.44	7.60	11.1	15.5
TAG-Unet [19]	2025	0.80	0.28	0.77	0.90	0.94	7.87	4.36	4.09	5.85	8.82
AMG [20]	2025	0.76	0.28	0.71	0.86	0.93	7.87	4.37	4.72	6.60	9.71
GDEA(Ours)	2026	**0.84**	**0.27**	**0.83**	**0.93**	**0.97**	**6.04**	**4.22**	**3.17**	**4.63**	**7.38**

GDEA is compared against state-of-the-art FEM surrogate models on additional datasets to validate its performance across diverse mechanical systems and mesh types. Table 2 compares the model's performance on the simulated additive manufacturing displacement fields dataset [19]. Our model achieves mixed results compared to GNOT [36] and outperforms all the other models. Compared to GNOT [36], our model achieves a higher mean and standard deviation of R^2, the same performance metrics in R^2 Q3 and the mean, standard deviation, and Q3 of RMSE, but lower performance in the median and Q1 of R^2 and the median of RMSE. While other differences are relatively marginal, our model reduces the standard deviation of R^2 by 49%, suggesting more consistent performance across meshes. Our model outperforms all other models across all criteria compared to state-of-the-art models, except for AMG [20] on R^2 Q3, where it is tied.

Table 2 Comparison to the state-of-the-art models on the simulated additive manufacturing displacement fields dataset [19].

Model	Year	R^2					RMSE				
		Mean (↑)	SD (↓)	Q1 (↑)	Median (↑)	Q3 (↑)	Mean (↓)	SD (↓)	Q1 (↓)	Median (↓)	Q3 (↓)
GINO [22]	2023	0.16	7.95	0.68	0.86	0.93	.014	.009	.008	.012	.017
GNOT [36]	2023	0.87	2.08	**0.93**	**0.98**	**0.99**	**.006**	**.006**	**.002**	**.004**	**.008**
Transolver [37]	2024	-0.14	14.5	0.64	0.82	0.91	.015	.010	.008	.013	.019
TAG-Unet [19]	2025	0.82	1.12	0.85	0.94	0.98	.009	.007	.004	.007	.011
AMG [20]	2025	0.86	1.74	0.89	0.97	**0.99**	.007	.008	.002	.005	.009
GDEA(Ours)	2026	**0.88**	**1.06**	0.92	0.97	**0.99**	**.006**	**.006**	.003	.005	**.008**

GDEA is compared against state-of-the-art FEM surrogate models on the flag-waving-in-the-wind dataset [32]. Table 3 shows that our model achieves mixed results

compared to multiple models. In R^2, our model achieves the same performance as GNOT [36], TAG-Unet [19], and AMG [20]. In terms of RMSE, our model achieves the same standard deviation as GNOT [36], TAG-Unet [19], and AMG [20], and the same Q3 as GNOT [36] and AMG [20]. The close agreement among the mean, Q1, median, and Q3, and the overall high performance, suggests that performance is near saturation, limiting differences between models and metrics.

Table 3 Comparison to the state-of-the-art models on the simulated flag-waving-in-the-wind dataset [32].

		R^2					RMSE				
Model	Year	Mean (↑)	SD (↓)	Q1 (↑)	Median (↑)	Q3 (↑)	Mean (↓)	SD (↓)	Q1 (↓)	Median (↓)	Q3 (↓)
GINO [22]	2023	0.93	0.11	0.93	0.97	0.98	0.21	0.08	0.16	0.19	0.25
GNOT [36]	2023	**0.98**	**0.02**	**0.98**	**0.99**	**0.99**	0.11	**0.03**	0.09	**0.10**	**0.12**
Transolver [37]	2024	0.91	0.16	0.90	0.95	0.98	0.24	0.10	0.17	0.22	0.29
TAG-Unet [19]	2025	**0.98**	**0.02**	**0.98**	**0.99**	**0.99**	0.11	**0.03**	0.09	0.11	0.13
AMG [20]	2025	**0.98**	**0.02**	**0.98**	**0.99**	**0.99**	**0.10**	**0.03**	**0.08**	**0.10**	**0.12**
GDEA(Ours)	2026	**0.98**	**0.02**	**0.98**	**0.99**	**0.99**	0.11	**0.03**	0.09	0.11	**0.12**

An ablation study is conducted on GDEA using Deformed ABC to understand the contribution of each component. Table 4 explains the contribution of the ablated components of the model. Ablating the DEQ component from the full model reduced the average R^2 by 4% and increased RMSE by 12%. Ablating the linear attention from the full model reduced the average R^2 by 13% and increased average RMSE by 39%. In contrast, comparing the model with and without Anderson acceleration shows that it has a limited impact on accuracy, yielding only minor, inconsistent differences. However, the results also show that the Anderson acceleration reduces execution time. Anderson acceleration reduced execution time by 30% in the model with linear attention and by 23% in the model without linear attention compared to the model without Anderson acceleration. This result is supported by the convergence iteration number demonstrated in Figure 2. The addition of average convergence iteration for node features (Figure 2 (a)) and DEQ gradients (Figure 2 (b)) is reduced by 40% with Anderson acceleration. Anderson acceleration reduced the number of iterations by 33% for node-feature convergence and by 50% for gradient convergence in the DEQ. Finally, Figure 2 (c) and (d) show that the number of convergence iterations is overall consistent across the attention blocks.

3 Discussion

On the Deformed ABC, Global-Weight Deep Equilibrium Attention (GDEA) achieves superior accuracy over state-of-the-art FEM surrogate models, particularly for finite systems, where prediction accuracy is generally lower, thereby demonstrating improved consistency. We attribute this result to our model mitigating under-reaching and over-smoothing, which can arise in complex and diverse graph data. However, the differences

Table 4 Ablation study within the GDEA on the Deformed ABC dataset.

Component			Time(sec)	R^2					RMSE				
Attention	DEQ	AA	Mean (↓)	Mean (↑)	SD (↓)	Q1 (↑)	Median (↑)	Q3 (↑)	Mean (↓)	SD (↓)	Q1 (↓)	Median (↓)	Q3 (↓)
	✓		.026	0.73	0.29	0.63	0.84	0.93	8.32	4.84	4.31	7.25	11.2
	✓	✓	.020	0.73	0.29	0.63	0.83	0.93	8.40	4.85	4.35	7.31	11.4
✓			.004	0.81	0.28	0.79	0.90	0.95	6.76	4.24	3.81	5.46	8.38
✓	✓		.030	**0.84**	**0.26**	**0.83**	**0.93**	0.96	6.10	**4.16**	3.25	4.75	7.53
✓	✓	✓	.021	**0.84**	0.27	**0.83**	**0.93**	**0.97**	**6.04**	4.22	**3.17**	**4.63**	**7.38**

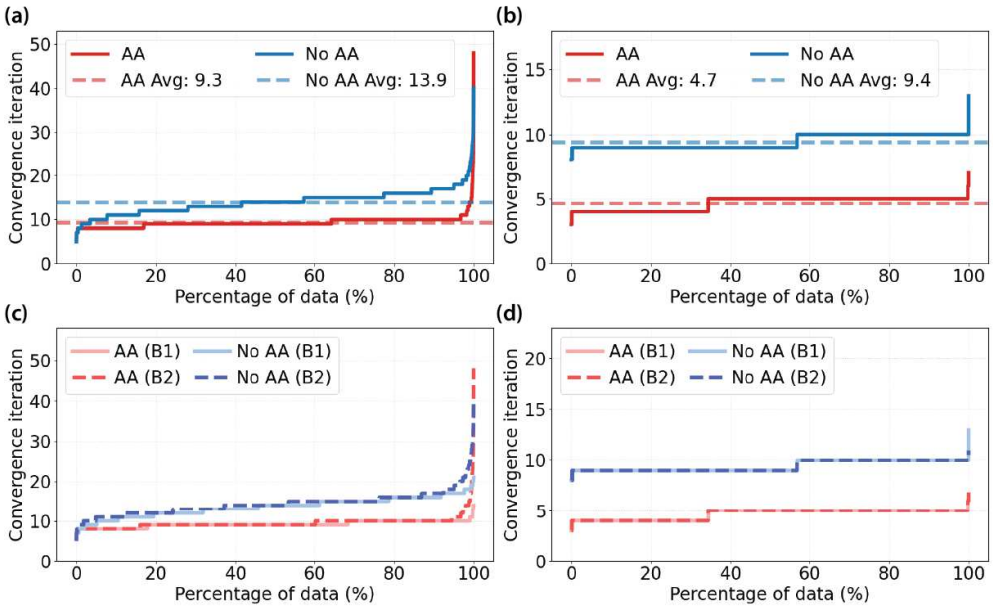


Fig. 2 This figure compares models with and without Anderson acceleration in terms of (a) the number of iterations required for latent node feature convergence and (b) the number of iterations required for gradient convergence. (c) and (d) show the convergence iterations of the latent node features and gradients for DEQs in different blocks.

are marginal and mixed on other datasets. This can be explained by the diversity of finite element scenarios. While Deformed ABC includes generic geometries, random materials, and random boundary conditions, the other two datasets contain more limited scenarios. The simulated additive manufacturing displacement fields dataset [19] consists of a single material and boundary condition, and the simulated flag-waving-in-the-wind dataset [32] includes a single geometry and material. As a result, these datasets are easier for surrogate models to predict, and they provide limited discrimination among surrogate models. Our model achieves comparable or better performance across diverse mechanical systems and mesh-type datasets, whereas other benchmark

models' performance varies by dataset, indicating that our model provides more consistent performance across different mechanical systems. Our model requires only 0.3% of the execution time of numerical FEM, achieving substantially higher computational efficiency with only a modest loss in accuracy. However, unlike one-shot prediction models, our model iteratively propagates information until the node features and the fixed-point gradients converge. Therefore, execution time may vary depending on data complexity. The ablation test results show that both the DEQ and the linear attention component improve prediction accuracy. These findings suggest that modeling node interactions through the global-weight matrix DEQ and capturing feature influence via linear attention are both effective. Among the two components, linear attention contributes more to prediction accuracy. This suggests that capturing the influence of node features on the solution is more important than modeling node interactions in finite-system surrogate modeling. Anderson acceleration does not improve prediction accuracy but reduces the number of iterations to convergence, thereby reducing execution time. However, the reduction in execution time and the number of iterations shows a discrepancy. The discrepancy between the execution-time reduction and the iteration reduction is attributed to other components of the execution, such as index matrix construction and neural network inference. The effect of Anderson acceleration is expected to become more pronounced as model complexity increases, leading to more iterations.

However, this study has several limitations. First, the model's predictions are not guaranteed to perfectly reflect real-world physical responses. Consequently, appropriate model usage and data selection rely on the user's domain expertise, as with other FEM tools. Second, the scalability of the proposed model may be limited for extremely large meshes, as it requires constructing and storing a global-weight matrix whose size scales with the number of mesh nodes. Third, experiments are conducted on a limited set of hyperparameters. The hyperparameters for each model are set to the optimized values obtained with Optuna [38]. While Optuna is widely used for hyperparameter optimization, alternative configurations may still lead to improved performance. Details of the hyperparameter selection process are provided in the Supplementary Material.

4 Method

4.1 Deformed ABC

We introduce Deformed ABC, a linear-elastic FEM dataset featuring diverse geometries, randomly selected materials, and boundary conditions. Geometries are represented as a mesh graph. FEM conditions and solutions are encoded as nodal features. Each node includes (1) three dimensional node coordinates, which represents position of the node, (2) Signed Distance Field (SDF), a negative distance of the node from the surface, (3) normal vector, (4) Dirichlet boundary condition, where zero is assigned to fixed nodes and one to free nodes, (5) Nodal loads, represented as three-dimensional vectors, corresponding to the force components, (6) material data, which is Young's modulus and Poisson's ratio, and (7) displacement, which is the solution of FEM. FEM surrogate models predict nodal displacements from other node-wise input features.

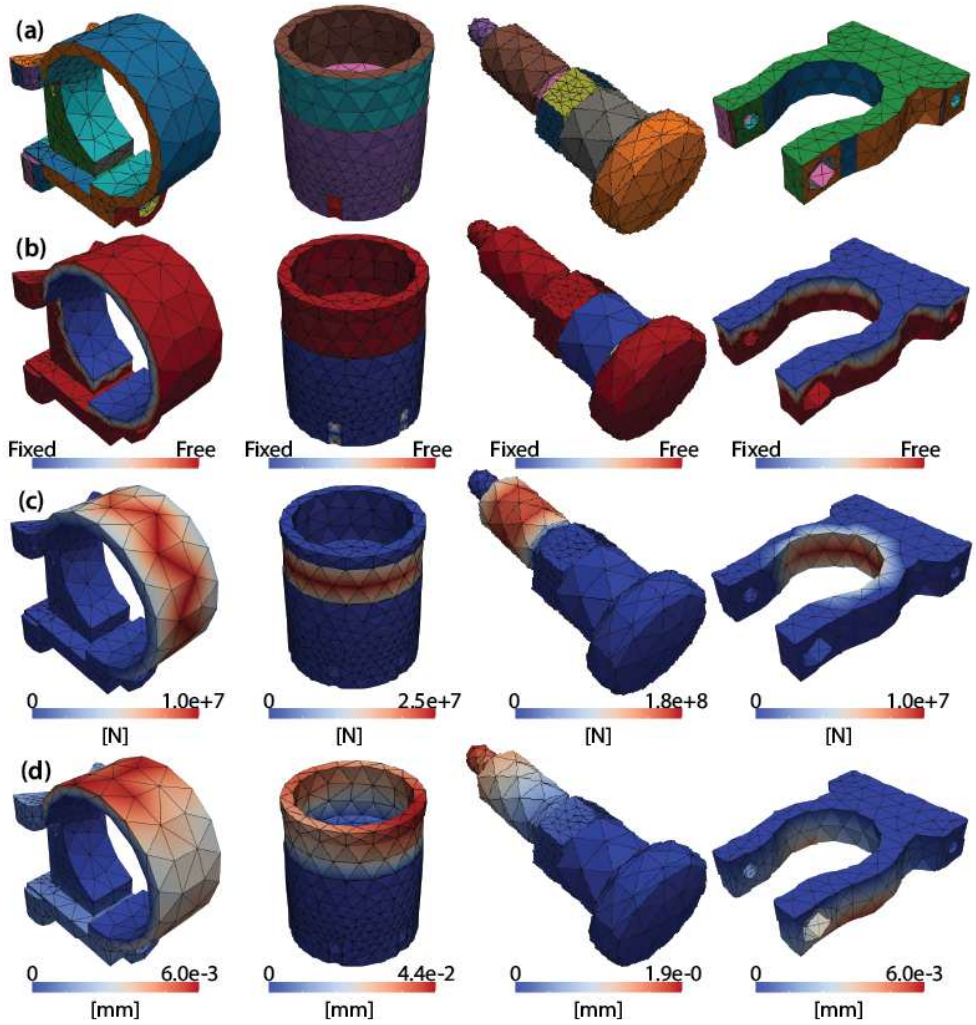


Fig. 3 This figure shows examples of different features of the mesh in Deformed ABC. (a) shows surface tags, inherited from the original CAD data. Different color indicates different surface tags. Surfaces for boundary conditions are randomly selected from these tags. (b) shows the Dirichlet boundary condition. (c) shows nodal load magnitude. (d) shows the magnitude of displacement, which is the solution of FEM from boundary conditions (b) and (c).

First, to collect generic geometries, we sampled them from the ABC dataset [35], which contains one million Computer-Aided Design (CAD) models for research in geometric deep learning. These CAD models are subsequently meshed into linear tetrahedral elements with a maximum element size of 10 mm using the Frontal-Delaunay algorithm for 2D elements [39] and the HXT algorithm for 3D elements [40]. Additionally, the mesh is optimized via Laplacian smoothing to refine the surface and edges and minimize distortion during meshing [41]. This maximum-size constraint and

optimization ensure that, even for larger geometric features, the element size remains appropriate, and the meshing algorithm automatically generates smaller elements in regions with finer geometric details [42]. Two surface areas on the mesh are randomly selected based on surface tags inherited from the original CAD model. Surfaces that cover more than 10% of the geometry are chosen to prevent trivial displacement or non-linear behavior. One is assigned as a fixed surface, also known as a Dirichlet boundary condition, and the other is designated as a loaded surface, also known as a Neumann boundary condition. Isotropic material is sampled from a wide range of material data to ensure data distribution that adequately represents all combinations of material data [43, 44]. Young's modulus is sampled from 0.01 GPa to 1,000 GPa on a log scale to cover a wide range of materials, including elastomers, foams, polymers, technical ceramics, composites, metals, and non-technical ceramics such as concrete, stone, brick, etc. [45]. Poisson's ratio is sampled from 0.01 to 0.5 for a wide range of material representations, including cork, glass, polymers, metals, rubber, etc. [46]. For the Neumann boundary condition, the direction of the load is randomly selected, and the traction magnitude is set to 0.1% of the shear modulus. Keeping the load magnitude fixed does not affect the generality of the results because the FEM solution scales linearly with the applied load under the assumption of linear elasticity. The nodal load is computed from the traction force, accounting for mesh resolution. Finally, the governing partial differential equation of linear elasticity is solved using FEM with linear shape functions. The resulting dataset is then preprocessed via filtering and scaling. Further details on the FEM formulation and data preprocessing are provided in the Supplementary Material. Figure 3 shows an example of surface tags, boundary conditions, and FEM solution.

4.2 GDEA: Global-Weight Deep Equilibrium Attention

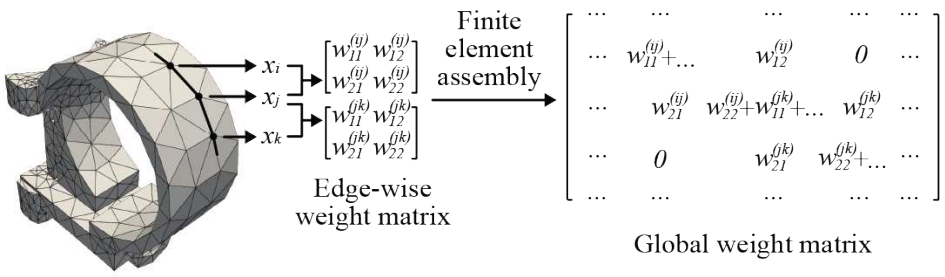


Fig. 4 This figure explains how the global-weight matrix is generated according to the mesh. First, the edge-wise weight matrix is predicted from nodal latent features. Then, edge-wise weight matrices are assembled into a global-weight matrix using the direct stiffness method.

The global-weight matrix deep equilibrium attention (GDEA) comprises multiple components. First, the nodal features are projected into a latent space via an encoder, a multi-layer perceptron (MLP) with residual connections. Then, the encoded embeddings are processed by multiple attention blocks, consisting of a global-weight matrix

deep equilibrium model (DEQ) followed by a linear self-attention layer, as illustrated in Figure 5.

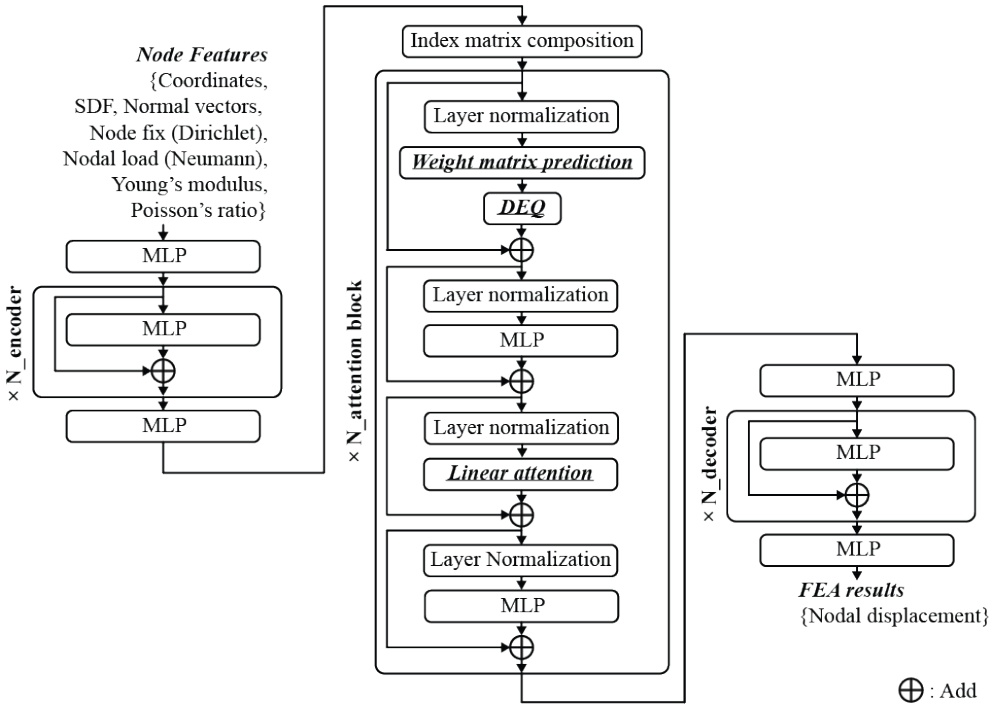


Fig. 5 This figure illustrates the architecture of our model, Global-Weight Matrix Deep Equilibrium Attention (GDEA). The encoder and decoder are implemented as MLPs with residual connections. The attention block consists of two main components: (1) a global-weight matrix DEQ that captures interactions between nodes, and (2) linear self-attention that models the influence of latent node features on the solution.

The global-weight matrix deep equilibrium model propagates information through multiple steps. First, the latent feature vectors of connected nodes are concatenated and fed into the MLP, which predicts the two-by-two edge-wise weight matrix $\mathbf{w}^{(ij)}$ following eq. 1, where x_i and x_j denote the node features of connected nodes i and j. The MLP includes scaled tanh re-projection to strictly enforce the maximum Lipschitz constant 0.9 and spectral normalization to ensure contraction mapping. Only three components of the edge-wise weight matrix are predicted, given its symmetry.

$$\mathbf{w}^{(ij)} = \begin{bmatrix} w_{11}^{(ij)} & w_{12}^{(ij)} \\ w_{12}^{(ij)} & w_{22}^{(ij)} \end{bmatrix} = \mathrm{MLP}(x_i, x_j), \tag{1}$$

Next, we assemble the global-weight matrix $\mathbf{W}$ from the edge-wise weight matrix $\mathbf{w}^{(ij)}$ following the direct stiffness method stated in eq. 2 [16]. $\mathbf{W}$ denotes the global stiffness matrix, $\mathbf{A}_{ij}$ is the local-to-global degree-of-freedom (DOF) mapping matrix,

and $\mathcal{E}$ is the set of edges. The size of the global-weight matrix is determined by the geometry's mesh size, allowing the model to handle varying mesh resolutions. The assembled global-weight matrix captures relationships among all nodes, including itself. Figure 4 explains how the global-weight matrix is generated according to the mesh.

$$\mathbf{W} = \sum_{(i,j)\in\mathcal{E}} \mathbf{A}_{ij}^{\top}\, \mathbf{w}^{(ij)}\, \mathbf{A}_{ij}, \quad \mathbf{w}^{(ij)} \in \mathbb{R}^{2\times 2}, \tag{2}$$

Afterward, each row of the global-weight matrix $\mathbf{W}$ is normalized to sum to 0.9. Restricting the maximum row sum to be less than 1 ensures that the mapping is a contraction. Ensuring that a mapping is a contraction is critical for two reasons. First, Banach's fixed-point theorem guarantees the existence and uniqueness of a fixed point for contraction mappings. Second, prior work shows that Anderson acceleration is r-linearly convergent when the fixed-point map is contractive, and the coefficients in the linear combination remain bounded [47]. The global-weight matrix $\mathbf{W}$ is used to iteratively update the node features following Eq. 3, where $\mathcal{N}(i)$ denotes the set of nodes connected to node i and $(\cdot)^{(k)}$ indicates the k-th iteration. Convergence is defined as the relative error in Eq. 4 falling below a predefined threshold, which is treated as a hyperparameter.

$$x_i^{(k+1)} = \sum_{j\in\mathcal{N}(i)} W_{ij}\, x_j^{(k)}, \tag{3}$$

$$\mathrm{RelErr}(z) = \frac{\|f(z) - z\|_2}{\|f(z)\|_2 + \epsilon}, \tag{4}$$

Accumulating gradients over iterations can easily lead to memory shortages. Accordingly, we employ a deep equilibrium model (DEQ) to efficiently compute the gradient via implicit differentiation [25]. The gradient $\frac{dz^*}{d\theta}$ is computed via a fixed-point iteration using Jacobian-vector products presented in eq. 5, which avoids the explicit inversion of the Jacobian matrix to significantly reduce the computational cost. z^* denotes the fixed point, θ represents the model parameters, $(\cdot)^{(k)}$ indicates the k-th iteration, and $\frac{\partial f(z^*,\theta)}{\partial z^*}$ denotes the Jacobian of f evaluated at the fixed point. $\left(\frac{dz^*}{d\theta}\right)^{(0)}$ is the upstream gradient of the model. Iteration stops when the relative error in Eq. 4 falls below a predefined threshold, which is treated as a hyperparameter.

$$\left(\frac{dz^*}{d\theta}\right)^{(k+1)} = \frac{\partial f(z^*,\theta)}{\partial\theta} + \frac{\partial f(z^*,\theta)}{\partial z^*}\left(\frac{dz^*}{d\theta}\right)^{(k)} \tag{5}$$

Anderson acceleration is used to accelerate the convergence of the iteration in equations eq. 3 and eq. 5. Anderson acceleration combines previous records to accelerate convergence, treating the number of records as a hyperparameter [26, 27, 47]. The optimal combination coefficients of the previous step are obtained by solving the normal equations with a Tikhonov regularization term 0.1 [48, 49]. A larger regularization term improves stability but slows convergence. Also, a damping factor of 0.5 is used, which defines the mixing ratio between Anderson-accelerated and naive updates

[27, 47]. A higher ratio of Anderson-accelerated results yields faster convergence speed, but decreases stability.

Subsequently, we incorporate linear attention to capture the influence of nodal latent features on the solution, as the global weight matrix focuses only on interactions among nodes. Linear attention reorders the computation of conventional attention to achieve linear computational cost, thereby enabling efficient learning on high-resolution data that would otherwise incur quadratic complexity with respect to the input size [24]. In the FEM context, linear attention first models global feature-solution relationships across all nodes and then applies them to individual nodes, effectively learning node-agnostic feature influence [23]. In this study, we implement a reduced form of the heterogeneous normalized linear attention proposed by Hao et al. [36]. While they separate node coordinates and physical features to compute cross-attention, we perform self-attention over a single latent space, thereby combining both spatial and physical data into a single latent representation. We combine these to ensure that the global-weight matrix captures the nodal relationships between the two feature types. Consequently, components that support the separation of nodal features, such as conditional embeddings and aggregation, are removed.

Equation 7 shows the normalized linear self-attention operation. Here, N denotes the sequence length, the number of nodes in the FEM context, D denotes the head feature dimension, and H denotes the number of heads. n indexes the target node, i indexes the source node, d and e index the head feature dimension ranging over 1 to D, h indexes the attention head, where n and e are free indices defined for all valid values of their range. $\tilde{Q}$ and $\tilde{K}$ are the element-wise transformed queries Q and keys K via a positive kernel feature map ϕ, as defined in 6.

$$\tilde{Q}^{(h)}_{n,d} = \phi\left(Q^{(h)}_{n,d}\right), \quad \tilde{K}^{(h)}_{i,d} = \phi\left(K^{(h)}_{i,d}\right), \quad \phi(x) = \mathrm{GELU}(x) + 1 \tag{6}$$

Q, K, and V are computed as $x_{\text{in}}W_Q$, $x_{\text{in}}W_K$, and $x_{\text{in}}W_V$, respectively, where x_{in} denotes the input feature and W_Q, W_K, and W_V are learnable weight matrices. The context is computed as the sum of values over all source nodes, and the attention output is normalized to ensure numerical stability, as shown in Equation 7 [36]. Afterwards, outputs from all heads are concatenated and projected using $\mathrm{concat}_{h=1}^{H}(\cdot)\,W_O$, where W_O is a learnable matrix and x_{out} represents the output of the linear self-attention layer. Finally, the linear attention output is fed into the decoder, which has the same architecture as the encoder, to map it to the nodal displacement and thereby predict the FEM solution.

$$x_{\text{out}} = \mathrm{concat}_{h=1}^{H}\left(\frac{\sum_{d=1}^{D}\tilde{Q}^{(h)}_{n,d}\sum_{i=1}^{N}\tilde{K}^{(h)}_{i,d}V^{(h)}_{i,e}}{\sum_{d=1}^{D}\tilde{Q}^{(h)}_{n,d}\sum_{i=1}^{N}\tilde{K}^{(h)}_{i,d}}\right)W_O \tag{7}$$

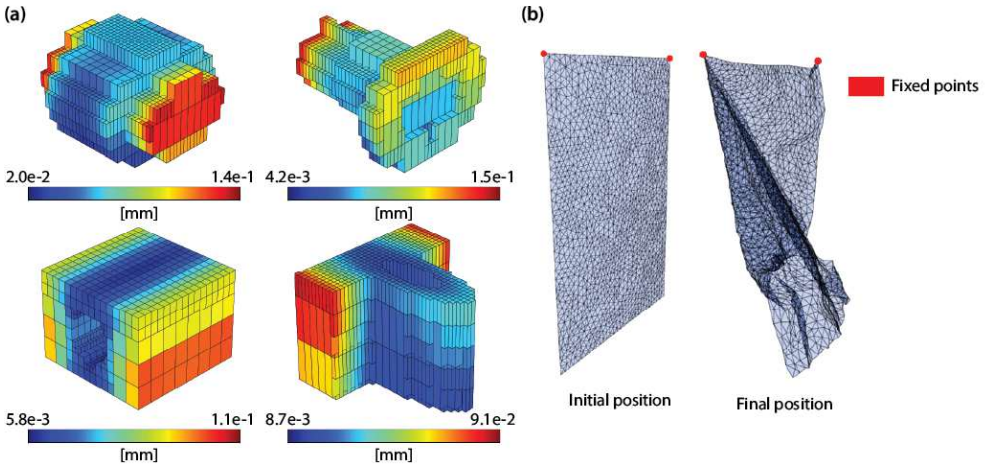


Fig. 6 This figure presents example data of two public datasets we used for benchmarking our model against state-of-the-art models. (a) presents examples of the additive manufacturing displacement fields dataset [19] and (b) presents an example of flag-waving in the wind dataset [32].

4.3 Experiment Setup

GDEA is compared against a variety of state-of-the-art FEM surrogate models across multiple datasets. We compare our model against GINO [22], GNOT [36], Transolver [37], TAG-U net [19], and AMG [20]. These models are trained and validated on three datasets: Deformed ABC, the additive manufacturing displacement fields dataset [19], and simulated flag-waving in the wind [32]. While Deformed ABC is newly introduced in this work, the other two datasets are previously published. The additive manufacturing displacement fields dataset [19] collects generic geometries from the Fusion 360 Gallery Segmentation Dataset [50] and simulates the assembly of each part via multiple finite element analyses, generating both thermal and mechanical histories during the build process. The example data is presented in 6 (a). Compared to Deformed ABC, this dataset conducts a more complex FEM but has lower diversity, as it includes only a single material and a single boundary condition. Also, the mesh consists of voxels of various sizes. For the additive manufacturing displacement fields dataset [19] experiment, the model's input is the coordinates of each node together with their connectivity, and its output is the displacement vector for each node. The simulated flag-waving in the wind dataset [32] simulates the position of a rectangular, meshed flag under wind conditions. The rectangular flag geometry, wind condition, and material stay the same, but the position of the flag and fixed points vary. This dataset includes the flag's position every 0.02 seconds over 400 steps, but we only use the initial and final positions to train and validate the model. The example data is presented in 6 (b). For the simulated flag-waving in the wind dataset [32] experiment, the model's input consists of the initial position coordinates, their connectivity, the node type (free or fixed), and the position of each node on the flag coordinate. The model's output is the final position of each node. Further Details of implementing these models on the datasets are provided in the Supplementary Material. Hyperparameters of all

models are optimized on each dataset using Optuna [38]. Details of the optimization and hyperparameters are presented in the Supplementary Material. Next, we conduct an ablation study to assess the contribution of each model component under the same settings as the comparative study on the Deformed ABC dataset.

Models are validated using R^2 and RMSE, widely used metrics for evaluating regression tasks [19, 42, 51]. The coefficient of determination R^2 quantifies the proportion of variance in the dependent variable explained by the independent variables, serving as a measure of model fit [51]. The Root Mean Squared Error (RMSE) is the square root of the mean of the squares of all the errors, indicating how close the prediction is to the ground truth [51]. To characterize performance across different geometric configurations, we analyze the distribution of these values using the mean, standard deviation, Q1, median, and Q3, providing insight into the variability of model performance across cases. All models were trained and validated on a workstation equipped with an AMD Ryzen Threadripper PRO 7965WX 24-core CPU, 256GB of ECC DDR5 RAM, and an NVIDIA RTX 6000 Ada Generation GPU (48GB VRAM).

Declarations

The authors declare no competing interests.

Data Availability

The new dataset used in this study is available on Hugging Face at huggingface.co/datasets/Junghunl/Deformed-ABC. Details of data generation are present in the paper and/or the Supplementary Material.

Code Availability

The code used in this study is available on GitHub at github.com/AiPEX-Lab. Details of code implementation are present in the paper and/or the Supplementary Material.

References

[1] Liu, W. K., Li, S. & Park, H. S. Eighty years of the finite element method: Birth, evolution, and future. *Arch Computat Methods Eng* **29**, 4431–4453 (2022).

[2] Thornton, J. The question of credibility. *Mechanical Engineering* **132**, 40–45 (2010).

[3] FONG, J. T. *et al.* Uncertainty in finite element modeling and failure analysis: A metrology-based approach. *Journal of pressure vessel technology* **128**, 140–147 (2006).

[4] Nath, D., Ankit, Neog, D. R. *et al.* Application of machine learning and deep learning in finite element analysis: A comprehensive review. *Arch. Computat. Methods Eng.* **31**, 2945–2984 (2024).

[5] Koeppe, A. *Deep learning in the finite element method*. Ph.D. thesis, Dissertation, Rheinisch-Westfälische Technische Hochschule Aachen, 2021 (2021).

[6] Xu, W., McComb, C. & Gutierrez, N. G. Taylor series error correction network for super-resolution of discretized partial differential equation solutions. *Journal of Computational Physics* **521**, 113569 (2025).

[7] Canziani, A., Paszke, A. & Culurciello, E. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678* (2016).

[8] Farmaga, I., Shmigelskyi, P., Spiewak, P. & Ciupinski, L. *Evaluation of computational complexity of finite element analysis*, 213–214 (IEEE, 2011).

[9] Skalicky, M., Pechanek, R., Sobotka, L. & Veg, L. *Impact of number of nodes used in 3d lptn on the results and computational time of the pmsm validated by fea and measurement*, 1–6 (IEEE, 2021).

[10] Ródenas, J., Bugeda, G., Albelda, J. & Onate, E. On the need for the use of error-controlled finite element analyses in structural shape optimization processes. *International journal for numerical methods in engineering* **87**, 1105–1126 (2011).

[11] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. & Monfardini, G. The graph neural network model. *IEEE transactions on neural networks* **20**, 61–80 (2008).

[12] Xu, K., Hu, W., Leskovec, J. & Jegelka, S. *How powerful are graph neural networks?* (2019). URL https://openreview.net/forum?id=ryGs6iA5Km.

[13] Li, Q., Han, Z. & Wu, X.-M. *Deeper insights into graph convolutional networks for semi-supervised learning*, Vol. 32 (2018).

[14] Jiang, B., Zhang, Z., Lin, D., Tang, J. & Luo, B. *Semi-supervised learning with graph learning-convolutional networks*, 11313–11320 (2019).

[15] Veličković, P. *et al. Graph attention networks* (2018). URL https://openreview.net/forum?id=rJz0nsA5F.

[16] Turner, M. J., Clough, R. W., Martin, H. C. & Topp, L. Stiffness and deflection analysis of complex structures. *journal of the Aeronautical Sciences* **23**, 805–823 (1956).

[17] Roth, A. & Liebig, T. *Rank collapse causes over-smoothing and over-correlation in graph neural networks*, 35–1 (PMLR, 2024).

[18] Li, G., Muller, M., Thabet, A. & Ghanem, B. *Deepgcns: Can gcns go as deep as cnns?*, 9267–9276 (2019).

[19] Ferguson, K. *et al.* Topology-agnostic graph u-nets for scalar field prediction on unstructured meshes. *Journal of Mechanical Design* **147**, 041701 (2025).

[20] Li, Z., Song, H., Xiao, D., Lai, Z. & Wang, W. *Harnessing scale and physics: A multi-graph neural operator framework for pdes on arbitrary geometries*, 729–740 (2025).

[21] Shehzad, A. *et al.* Graph transformers: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2026).

[22] Li, Z. *et al.* Geometry-informed neural operator for large-scale 3d pdes. *Advances in Neural Information Processing Systems* **36**, 35836–35854 (2023).

[23] Shen, Z., Zhang, M., Zhao, H., Yi, S. & Li, H. *Efficient attention: Attention with linear complexities*, 3531–3539 (2021).

[24] Katharopoulos, A., Vyas, A., Pappas, N. & Fleuret, F. *Transformers are rnns: Fast autoregressive transformers with linear attention*, 5156–5165 (PMLR, 2020).

[25] Bai, S., Kolter, J. Z. & Koltun, V. Deep equilibrium models. *Advances in neural information processing systems* **32** (2019).

[26] Anderson, D. G. Iterative procedures for nonlinear integral equations. *Journal of the ACM (JACM)* **12**, 547–560 (1965).

[27] Walker, H. F. & Ni, P. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis* **49**, 1715–1735 (2011).

[28] Bonnet, F., Mazari, J., Cinnella, P. & Gallinari, P. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. *Advances in Neural Information Processing Systems* **35**, 23463–23478 (2022).

[29] Regenwetter, L., Weaver, C. & Ahmed, F. Framed: An automl approach for structural performance prediction of bicycle frames. *Computer-Aided Design* **156**, 103446 (2023).

[30] Hong, S., Kwon, Y., Shin, D., Park, J. & Kang, N. Deepjeb: 3d deep learning-based synthetic jet engine bracket dataset. *Journal of Mechanical Design* **147**, 041703 (2025).

[31] Elrefaie, M. *et al.* Ai agents in engineering design: a multi-agent framework for aesthetic and aerodynamic car design. *arXiv preprint arXiv:2503.23315* (2025).

[32] Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A. & Battaglia, P. *Learning mesh-based simulation with graph networks* (2020).

[33] Regenwetter, L., Nobari, A. H. & Ahmed, F. Deep generative models in engineering design: A review. *Journal of Mechanical Design* **144**, 071704 (2022).

[34] Sanchez-Gonzalez, A. *et al. Learning to simulate complex physics with graph networks*, 8459–8468 (PMLR, 2020).

[35] Koch, S. *et al. Abc: A big cad model dataset for geometric deep learning*, 9601–9611 (2019).

[36] Hao, Z. *et al. Gnot: A general neural operator transformer for operator learning*, 12556–12569 (PMLR, 2023).

[37] Wu, H., Luo, H., Wang, H., Wang, J. & Long, M. Salakhutdinov, R. *et al.* (eds) *Transolver: A fast transformer solver for PDEs on general geometries.* (eds Salakhutdinov, R. *et al.*) *Proceedings of the 41st International Conference on Machine Learning*, Vol. 235 of *Proceedings of Machine Learning Research*, 53681–53705 (PMLR, 2024). URL https://proceedings.mlr.press/v235/wu24r.html.

[38] Akiba, T., Sano, S., Yanase, T., Ohta, T. & Koyama, M. *Optuna: A next-generation hyperparameter optimization framework* (2019).

[39] Rebay, S. Efficient unstructured mesh generation by means of delaunay triangulation and bowyer-watson algorithm. *Journal of computational physics* **106**, 125–138 (1993).

[40] Marot, C., Pellerin, J. & Remacle, J.-F. One machine, one minute, three billion tetrahedra. *International Journal for Numerical Methods in Engineering* **117**, 967–990 (2019).

[41] Schöberl, J. Netgen an advancing front 2d/3d-mesh generator based on abstract rules. *Computing and visualization in science* **1**, 41–52 (1997).

[42] Ezemba, J., McComb, C. & Tucker, C. Neural network surrogate modeling for stochastic finite element method using three-dimensional graph representations: A comparative study. *Journal of Mechanical Design* **148**, 011704 (2026).

[43] McKay, M. D., Beckman, R. J. & Conover, W. J. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* **42**, 55–61 (2000).

[44] Helton, J. C. & Davis, F. J. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System Safety* **81**, 23–69 (2003).

[45] Ashby, M. F. & Cebon, D. Materials selection in mechanical design. *Mrs Bull* **30**, 995 (2005).

[46] Greaves, G. N., Greer, A. L., Lakes, R. S. & Rouxel, T. Poisson's ratio and modern materials. *Nature materials* **10**, 823–837 (2011).

[47] Toth, A. & Kelley, C. T. Convergence analysis for anderson acceleration. *SIAM Journal on Numerical Analysis* **53**, 805–819 (2015).

[48] Shi, W. *et al.* Regularized anderson acceleration for off-policy deep reinforcement learning. *Advances in Neural Information Processing Systems* **32** (2019).

[49] Golub, G. H., Hansen, P. C. & O'Leary, D. P. Tikhonov regularization and total least squares. *SIAM journal on matrix analysis and applications* **21**, 185–194 (1999).

[50] Lambourne, J. G. *et al. Brepnet: A topological message passing system for solid models*, 12773–12782 (2021).

[51] Tatachar, A. V. Comparative assessment of regression models based on model evaluation metrics. *International Research Journal of Engineering and Technology (IRJET)* **8**, 2395–0056 (2021).

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- GDEAGlobalWeightDeepEquilibriumAttentionSupplementaryMaterials.pdf