Advanced Neural Network for Surface Reconstruction with Applications in Medical Imaging

Amir Noorizadegan[*], D.L. Young [†*§], C.S. Chen[‡], Chuin-Shan Chen[*§]

Abstract

Surface reconstruction from point clouds is a fundamental challenge in computer graphics and medical imaging. In this paper, we explore the application of advanced neural network architectures for the accurate and efficient reconstruction of surfaces from data points. We introduce a novel variant of the Residual Network (ResNet) called Square-ResNet (SQR-ResNet) and investigate its performance alongside plain neural networks and standard ResNet in various numerical examples. These examples include the reconstruction of simple and complex surfaces, such as spheres, human hands, and intricate models like the Stanford Bunny. We analyze the impact of factors such as the number of hidden layers, interior and exterior points, and data distribution on surface reconstruction quality. Our results show that the proposed SQR-ResNet architecture outperforms other neural network configurations, achieving faster convergence and higher-quality surface reconstructions. Additionally, we demonstrate the SQR-ResNet's ability to predict surfaces over missing data, a valuable feature for challenging applications like medical imaging. This research contributes to the field of machine learning in computer graphics and medical imaging, offering new insights and methods for accurate and efficient surface reconstruction.

Keywords: Surface reconstruction, point clouds, Deep learning ,computer graphic, medical imaging.

1 Introduction

Computer graphics is a rapidly evolving field that continually seeks innovative techniques for surface reconstruction and visualization, offering a broad spectrum of applications from 3D object modeling to animation, rendering, and beyond [1]. Surface reconstruction from point clouds has been a subject of considerable research effort, where methods

[*]Department of Civil Engineering, National Taiwan University, 10617, Taipei, Taiwan
[†]Core Tech System Co. Ltd, Moldex3D, Chubei, Taiwan
[‡]School of Mathematics and Natural Sciences, University of Southern Mississippi, USA
[§]Corresponding authors: `dchen@ntu.edu.tw`, `dlyoung@ntu.edu.tw`

are categorized into those that employ data-driven priors and those that do not. The latter class, including methods like Method of foundanamnetla solutions [2], radial basis function method [3], scale space meshing [4] and Ohrhallinger's combinatorial approach [5], offers effective solutions but often struggles with noisy point clouds or non-smooth surfaces. Additionally, deformations and patch-based methods [6, 7] prove to be limited in handling complex topologies and connectivity changes. Poisson reconstruction [8, 9] stands as the prevailing benchmark in non-data-driven surface reconstruction from point clouds. Notably, none of the previously mentioned methods employ a prior that distills information about typical surface shapes from a vast dataset.

In contrast, data-driven methods have emerged as potent solutions, deriving priors from extensive datasets. These methods, including AtlasNet [10], Scan2Mesh [11], Points2surf [1], and others, leverage feature representations to adeptly handle noisy or partial input. While these algorithms necessitate extensive datasets and a latent feature space, with a primary focus on encoding shapes that fall within this feature space, our approach charts a distinctive course. Our method is founded upon feedforward neural networks, employing an interpolation approach. This simple architectural design is specifically crafted to address the challenges posed by both smooth and non-smooth 3D domains. This divergence in architecture sets our approach apart, promising heightened computational capabilities for a wide array of surface reconstruction scenarios. Notably, our method has already demonstrated its strengths in our recent work addressing interpolation and inverse problems [12]. Our research objectives span three key aspects:

1. **Introduction of the "Power-Enhancing ResNet" architecture**: This paper presents a robust neural network architecture designed to meet the computational demands of computer graphics challenges.

2. **Surpassing conventional plain neural networks**: Extensive experiments demonstrate the superior performance of our architecture when compared to traditional plain neural networks.

3. **Exploration of diverse computer graphics scenarios**: Our research encompasses a diverse range of computer graphics scenarios, enhancing our understanding of the application of this innovative architecture.

Our rigorous comparisons empirically demonstrate the manifold advantages of our proposed architecture within the domain of computer graphics, highlighting its potential to address challenging surface reconstruction problems effectively.

The subsequent sections of this paper, section 2 delves into neural networks and their application to solve computer graphics problems. Section 3 explores the architecture of residual networks and introduces the innovative Power-Enhancing ResNet. Here, we elucidate the incorporation of power terms and underscore their potential benefits. Section 4 provides a detailed account of our experimental setup, the evaluation of results, and an in-depth discussion of our findings. Section 5 concludes the paper by summarizing our contributions and offering insights into potential directions for future research.

2 Neural Networks in Computer Graphics

In this section, we delve into the application of feedforward neural networks for solving complex problems in the field of computer graphics. Specifically, we focus on the intricate task of constructing surfaces from point cloud data, a critical component in computer graphics and 3D modeling.

2.1 Feedforward Neural Networks

The feedforward neural network, also commonly referred to as a multilayer perceptron (MLP), serves as a foundational architectural framework within the realm of artificial neural networks. These networks are characterized by their interconnected layers of neurons, where the flow of information moves unidirectionally, starting from the input layer and traversing through various hidden layers before reaching the output layer. This unidirectional data flow, often described as "feedforward," plays a crucial role in transforming input data into precise and valuable output predictions.

At the core of a feedforward neural network lie its individual neurons. Each neuron is tasked with calculating a weighted sum of its inputs, which is subsequently modified by the inclusion of a bias term. Following this summation, an activation function is applied to the result. For a neuron situated in a layer i with n input data points,

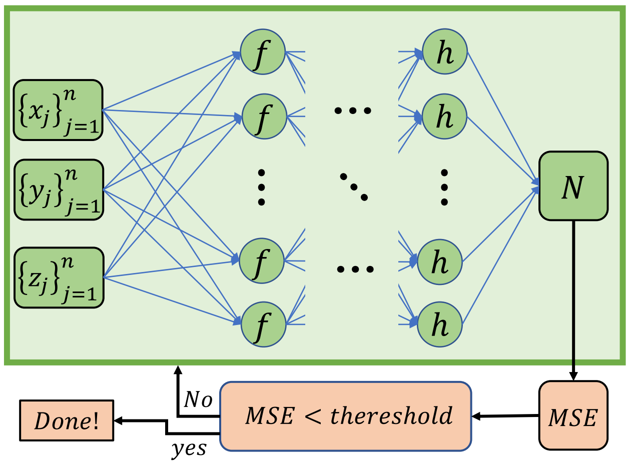


Figure 1: A Feedforward neural network architecture.

where these inputs are represented as $\mathcal{P} = [\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n]$, the respective weights as $\mathbf{w} = [w_1, w_2, \ldots, w_n]$, and the bias as b_i, the output z_i is determined as follows:

$$z_i = \sum_{j=1}^{n} w_j \mathbf{p}_j + b_i \tag{1}$$

Here, $\mathbf{p}_j$ exists in a 3-dimensional space, represented as $\mathbf{p} = (x, y, z)$. Subsequently, the output undergoes a transformation using an activation function $h(\cdot)$:

$$y_i = h(z_i) \tag{2}$$

It is crucial to note that while hidden layers make use of activation functions to introduce non-linearity, the final layer, typically referred to as the output layer, often does not employ an activation function for its outputs. Figure 1 provides a schematic representation of a feedforward neural network designed for function interpolation, with f defined as follows:

$$f(\mathbf{p}) = h\left(\sum_{j=1}^{n} w_j \mathbf{p}_j + b\right) \tag{3}$$

This represents a combination of the linear transformation (1) and the nonlinear activation function (2).

2.1.1 Mathematical Foundation

In this section, we provide a mathematical foundation for the proposed method that leverages neural networks for surface generation in computer graphics. Let $\mathcal{D}$ represent the 3D domain, and let $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ denote the set of data points sampled within this domain. Each data point $\mathbf{p}_j$ has spatial coordinates $\mathbf{p}_j = (x_j, y_j, z_j)$. We categorize the data points into three sets:

$$\begin{aligned}
\mathcal{P}_{\text{interior}} &: \text{Data points located within the 3D object} \\
\mathcal{P}_{\text{surface}} &: \text{Data points defining the surface boundary} \\
\mathcal{P}_{\text{exterior}} &: \text{Data points outside the object}
\end{aligned}$$

We define n_i, n_s, and n_e as the number of points inside, on the surface, and outside of the domain $\mathcal{D}$. We also consider n as the total number of data points, where $n = n_i + n_s + n_e$.

To train the neural network, we label each data point $\mathbf{p}_j$ as follows:

$$\text{Label}(\mathbf{p}_j) = \begin{cases} 1 & : \mathbf{p}_j \in \mathcal{P}_{\text{interior}} \\ 0 & : \mathbf{p}_j \in \mathcal{P}_{\text{surface}} \\ -1 & : \mathbf{p}_j \in \mathcal{P}_{\text{exterior}} \end{cases} \tag{4}$$

The neural network architecture consists of a combination of feedforward and residual networks, with the power-enhanced version. It is designed to learn the distinct features associated with each category of data points. The loss function, often chosen as the mean squared error (MSE), quantifies the difference between predicted labels and actual labels. It is expressed as:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} \left(\text{Label}(\mathbf{p}_i) - \text{Prediction}(\mathbf{p}_i)\right)^2 \tag{5}$$

where n is the number of data points, Label($\mathbf{p}_i$) represents the ground truth label, and Prediction($\mathbf{p}_i$) is the network's predicted label for data point $\mathbf{p}_i$. The network's parameters, including weights and biases, are optimized using an iterative optimization algorithm, L-BFGS-B (Limited-memory Broyden-Fletcher-Goldfarb-Shanno with Box constraints), to minimize the MSE. This process equips the network with the ability to accurately predict labels for new data points and, subsequently, to generate surfaces for 3D objects in computer graphics. Two noteworthy points merit consideration:

- When utilizing interpolation and assigning value labels of -1, 0, and 1, it is expected that during the interpolation process, some input points will naturally yield a label of 0, indicating their location on the surface.
- Our proposed method necessitates the availability of either interior or exterior points or both, even when data on the surface is the only information present. These interior and exterior points can be determined through various means, such as manual identification or using algorithms like MATLAB's `pcnormals`.

2.1.2 Testing and Surface Generation

After training the neural network, the next step is to apply the model to generate surfaces for 3D objects. This involves testing the network's ability to predict labels for a set of testing data points. In our case, we employ a grid-based approach to define the testing data points over the 3D domain $\mathcal{D}$. Let $\mathcal{T} = \{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_m\}$ represent the set of testing data points, where each point $\mathbf{t}_j$ corresponds to a meshgrid point over the 3D domain. We use these points to evaluate the network's performance in surface generation.

To visualize the surfaces, we utilize the Matlab `‘‘isosurface’’` command. This command allows us to create contours over the 3D domain based on the predicted labels from the neural network. Specifically, we set the isosurface threshold value to "0" effectively marking the boundary of the generated surface. In this way, we obtain a 3D contour that visually represents the generated surface for the 3D object. By selecting a threshold of "0" we ensure that the contour encircles the surface points, which were labeled as "0" during the training process (4).

The Matlab-generated contour provides a visual verification of the surface generated by our method, which is particularly valuable for computer graphics applications. The successful creation of these contours demonstrates the effectiveness of our power-enhanced neural network in approximating and generating surfaces for 3D objects. This approach offers a practical and intuitive way to assess the quality of the surfaces generated by the proposed method, making it a valuable tool in computer graphics for 3D object visualization.

3 Residual Network

Residual networks, commonly recognized as ResNets [13, 14], have emerged as a dominant architectural paradigm in neural networks. They are notably distinguished by their residual modules, denoted as f_i, and the integration of skip connections to bypass these modules, enabling the assembly of deep networks. This construction facilitates the creation of residual blocks, constituting a cluster of layers within the network [15,16]. Unlike the depiction of the basic neural network in Fig. 2(a), Fig. 2(b) illustrates the incorporation of ResNet attributes into the network architecture. To simplify our discourse, we shall omit discussions of the initial pre-processing and final steps. Consequently, the definition of the output g_i for the i-th layer is as expressed below:

$$g_i = f_i(g_{i-1}) + g_{i-1}. \tag{6}$$

where f is defined in (3). In this research, we introduce an innovative approach known as the "SQR-ResNet." This variant is characterized by an adjusted recursive definition of the form:

$$\begin{cases} g_i = f_i(g_{i-1}) + g_{i-1}^2, & \text{for} \quad i = 1, 3, 5, \ldots \\ g_i = f_i(g_{i-1}), & \text{for} \quad i = 2, 4, 6, \ldots \end{cases} \tag{7}$$

This distinctive configuration, portrayed in Fig. 2(c), introduces the incorporation of a power term, g_{i-1}^2, for specific layers, augmenting the network's expressive capabilities.

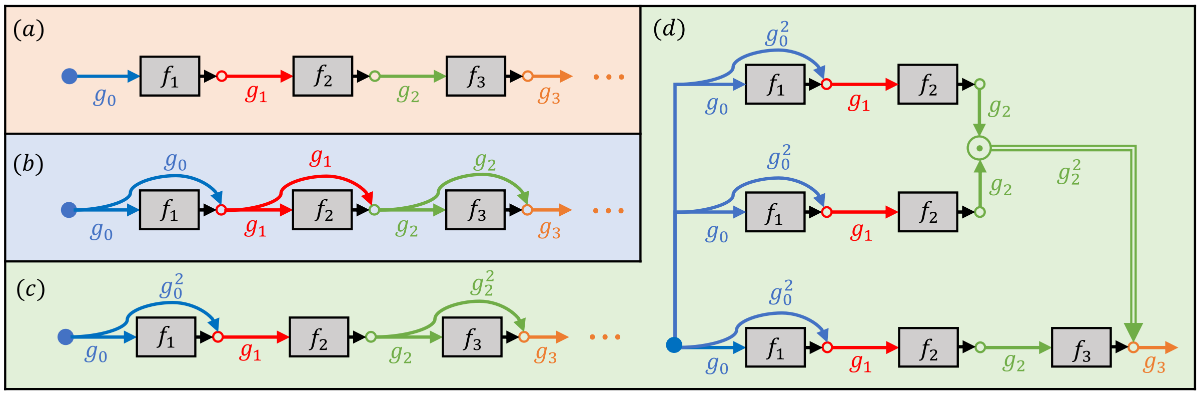


Figure 2: Three neural network architectures: (a) plain neural network (Plain NN), (b) residual network (ResNet), (c) SQR-ResNet, and (d) Unraveled SQR-ResNet where $\odot$ denotes element-wise multiplication.

Figure. 2 offers a visual representation of three neural network architectures: (a) the plain neural network (Plain NN), (b) the residual network (ResNet), (c) the SQR-ResNet, and (d) the Unraveled SQR-ResNet. Here, the symbol $\odot$ signifies element-wise multiplication. To facilitate comparative assessment among Plain NN, ResNet, and SQR-ResNet (portrayed in Figs. 2(a)-(c), respectively), we evaluate the output of the third hidden layer with respect to the input $g_0 = \mathcal{P}$. The outcomes for the plain neural network are outlined below:

$$\begin{aligned} g_3 &= f_3(g_2) \\ &= f_3(f_2(g_1)) \\ &= f_3(f_2(f_1(g_0))) \end{aligned} \tag{8}$$

In contrast, the corresponding ResNet formulation, as articulated by Veit et al. [17], can be summarized as:

$$\begin{aligned} g_3 &= f_3(g_2) + g_2 \\ &= f_3(f_2(g_1) + g_1) + [f_2(g_1) + y_1] \\ &= f_3(f_2(f_1(g_0) + g_0) + f_1(g_0) + g_0) + [f_2(f_1(g_0) + g_0) + f_1(g_0) + g_0] \end{aligned} \tag{9}$$

Finally, the formulation for the initial three hidden layers within the SQR-ResNet is defined as follows:

$$\begin{aligned} g_3 &= f_3(g_2) + g_2^2 \\ &= f_3(f_2(g_1)) + [f_2(g_1)]^2 \\ &= f_3(f_2(f_1(g_0) + g_0^2)) + \left[f_2(f_1(g_0) + y_0^2)\right]^2 \end{aligned} \tag{10}$$

Figure 2(d) visually represents the intricate "expression tree," offering a comprehensive illustration of the data flow from input to output. This graphical depiction underscores the existence of diverse data pathways that facilitate or bypass particular residual modules.

4 Numerical Analysis

In the context of this study, we make use of the notations n, n_l, and n_n to describe the count of data points, layers, and neurons in each layer, respectively. Additionally, n_i, n_s, n_e, and n_t are employed to denote the number of interior, surface, exterior, and validation data points.

In this section, an examination of four distinct approaches is undertaken:

1. `Plain NN`: A standard neural network lacking additional modifications or residual connections (refer to Fig. 2(a)).

2. `ResNet`: This neural network architecture incorporates residuals, which are added to the output of every other layer.

3. `SQR-ResNet`: An innovative variation of the Residual NN architecture, introducing squared residuals every other layer. This method involves obtaining the output of every alternate layer by squaring the preceding layer's output and subsequently adding the squared residual (depicted in Fig. 2(c)-(d)).

In all following examples, we exclusively considered interior points and the point clouds on the surface. In the last example, we incorporated both interior and exterior points for the surface reconstruction.

Example 1 In our first numerical example, we examine the construction of a simple and smooth surface, a sphere. The sphere has a radius of 1 and is centered at the origin (0, 0, 0), as depicted in Figure 3. We employ 200 data points on the sphere's surface (n_s), along with 20 interior points (n_i), without any exterior points. The points are randomly selected. Figure 4 displays the results of the surface construction over these points. The green surfaces represent the outputs of the neural network, while the red meshed spheres are provided for reference to enhance the visualization of the results.

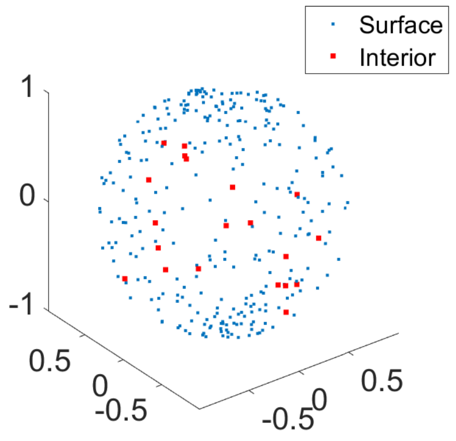


Figure 3: The profile of the point distribution for the sphere.

The first row of Figure 4 illustrates the results using the Plain NN for different epochs, ranging from 10 to 400. The second row showcases the results for the same epochs, employing the ResNet approach. Finally, the last row demonstrates the results for the proposed SQR-ResNet.

Several observations can be made from this example:

- The proposed SQR-ResNet demonstrates superior performance at specific iterations, especially when compared to the Plain NN. In other words, it converges faster when evaluating the constructed surface at specific epochs compared to the other networks.
- The ResNet exhibits better performance than the Plain NN.
- At the final epoch, we observe that the SQR-ResNet is very close to the reference sphere, while the other two networks have not yet achieved an accurate representation.

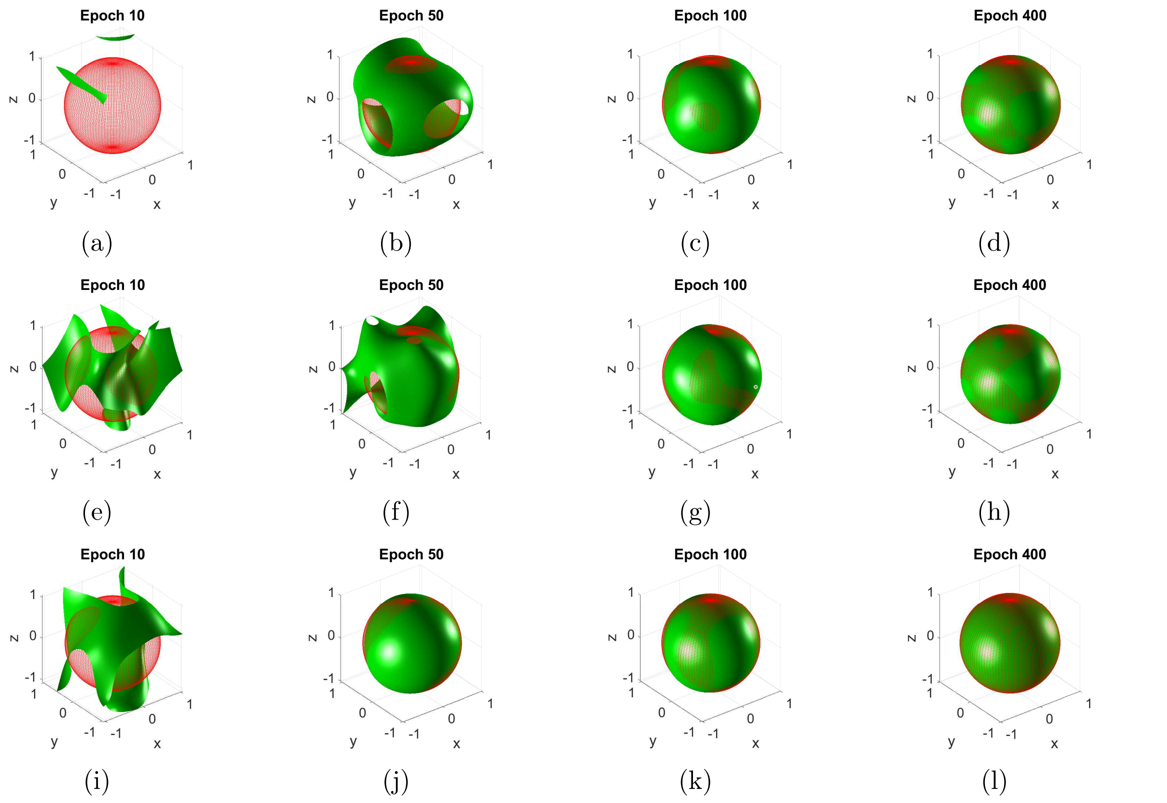


Figure 4: The profile of the simulated sphere. The top panel: Plain NN, middle panel: ResNet, bottom panel: SQR-ResNet. Font size changed.

In conclusion, the proposed method outperforms the other approaches. Next, we investigate the impact of the number of interior points (n_i) and the number of neurons in each hidden layer (n_n) in our work. For this analysis, we exclusively consider the proposed SQR-ResNet. Figure 5 displays the results of constructing the sphere in 2D for various numbers of interior points (top row) and the number of hidden layers (bottom row).

In the top row, starting with just one interior point (plot 5(a)), we observe that the proposed method performs well even with this minimal number of interior points. As we increase the number of interior points to 100 (plot 5(b)), we notice that the sphere's edge becomes less smooth. After selecting 200 interior points (plot 5(c)), in the top-right corner, we observe that the edges become even less smooth, with some wave-like artifacts appearing on the sphere's surface.

We maintain the settings at 200 interior points and 5 hidden layers and vary the number of neurons (n_n) to examine this parameter's impact on the constructed surface, as shown in the bottom row. As we increase the number of neurons from 50 (plot 5(c)) to 100 (plot 5(d)), we observe that the sphere's surface becomes smoother. Further increasing the number of neurons to 200 (plot 5(e)) and finally to 300 (plot 5(f)) results in a notably smooth and improved surface.

A comparison between the top-right (plot 5(c)) and bottom-right corners (plot 5(f))

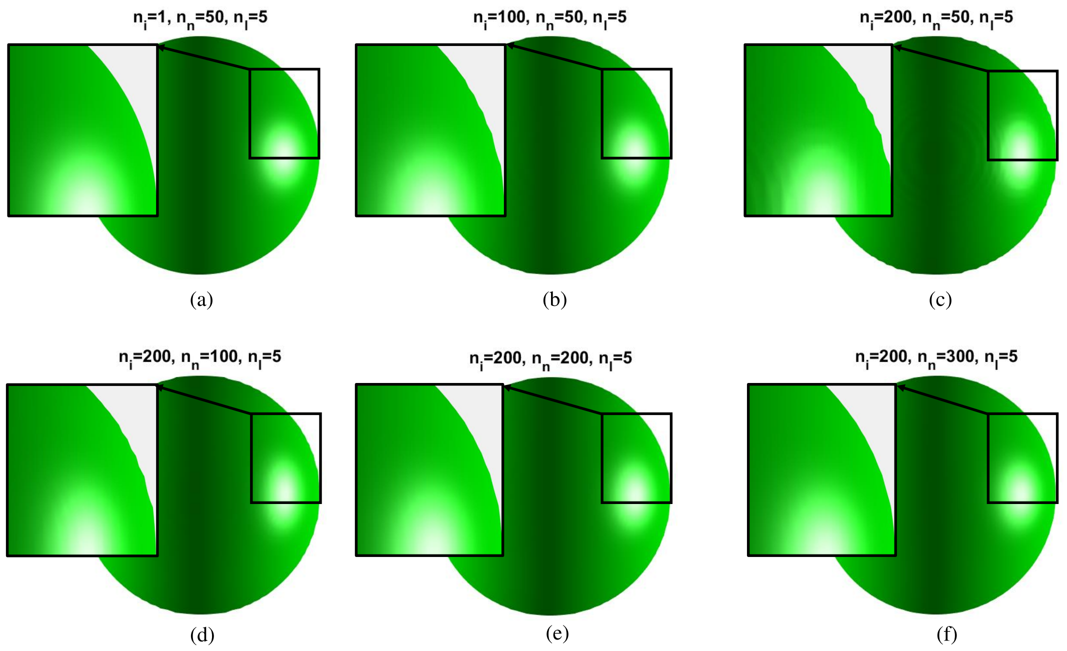

n_i=1, n_n=50, n_l=5
n_i=100, n_n=50, n_l=5
n_i=200, n_n=50, n_l=5
(a)
(b)
(c)
n_i=200, n_n=100, n_l=5
n_i=200, n_n=200, n_l=5
n_i=200, n_n=300, n_l=5
(d)
(e)
(f)

Figure 5: The profile of the simulated sphere for $n = 200$ using SQR-ResNet for various number of interior points (top panel) and neurons (bottom panel.)

demonstrates that increasing the number of neurons is beneficial for constructing a smoother and more accurate surface. However the selection of the interior points should be done by care.

Example 2 In this example, we tackle a more intricate surface, a hand. Figure 6 illustrates a node distribution where points on the hand's surface are depicted in blue, and the smaller interior points are shown in red.

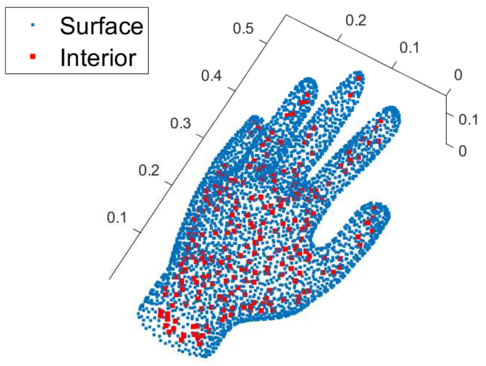


Figure 6: The profile of the point distribution for the hand.

Moving on to the numerical results, Figure 7 showcases the outcomes of three different network structures for 3000 points on the surface, with 500 interior points. The structure of the network comprises five hidden layers, each containing 50 neurons. The top panel corresponds to the Plain Neural Network, the middle panel displays the results of the ResNet algorithm, and the bottom panel exhibits the results of our proposed SQR-ResNet. Within each panel, columns represent different Epoch numbers, specifically 50, 5000, and 10000 Epochs, facilitating easy comparison of convergence across the three network architectures. The final column, Column 4, displays the results at the convergence point.

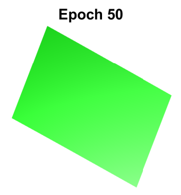
Several observations can be made from this figure:

- At a small number of Epochs, such as 50 as depicted in the first column, the construction begins as a simple plate for all three network structures.
- Subsequently, better convergence can be observed with an increasing number of Epochs, particularly for ResNet and SQR-ResNet, in contrast to the Plain Neural Network. For example, a comparison between plots in Figure 7(c,g,k) reveals the emergence of a recognizable hand shape for ResNet and SQR-ResNet, while the Plain Neural Network produces non-meaningful results.
- The final results, shown in the last column, reveal a near-perfect hand shape for SQR-ResNet, as seen in Figure 7(l). In contrast, the Plain Neural Network's results, as illustrated in Figure 7(d), are highly polluted, and ResNet's outcomes, Figure 7(h), feature some extraneous surface artifacts around the hand.
- Notably, the Plain Neural Network requires a substantially smaller number of Epochs for convergence (13250), while ResNet exhibits the longest convergence duration at 21100 Epochs. Conversely, SQR-ResNet achieves its final results at Epoch 18000, positioning it between the Plain Neural Network and ResNet in terms of the number of required Epochs.

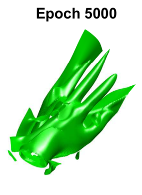
Next, we investigate the impact of varying the number of hidden layers in this example. Figure 8 illustrates the results obtained when altering the number of hidden layers, ranging from 3 to 5, 10, and finally 15 hidden layers. These experiments were conducted with the same settings as the previous figure, with 100 neurons allocated for each hidden layer.

Several key observations can be made:

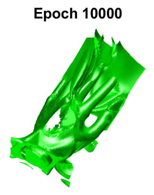
- Utilizing only three hidden layers results in a hand shape that hasn't fully formed yet.
- Conversely, employing a larger number of hidden layers, such as 15, introduces additional surfaces.
- The configurations with 5 and 10 hidden layers appear to work well in generating accurate results.



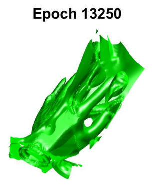
(a)



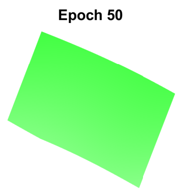
(b)



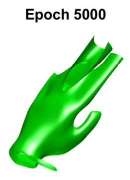
(c)



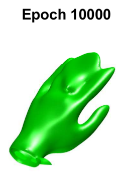
(d)



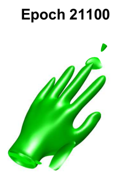
(e)



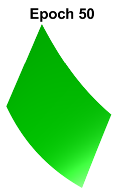
(f)



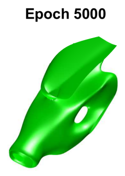
(g)



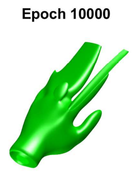
(h)



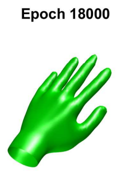
(i)



(j)



(k)

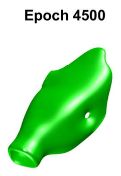


(l)

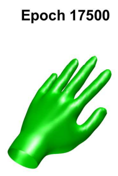
Figure 7: The profile of the simulated hand. The top panel: Plain NN, middle panel: ResNet, bottom panel: SQR-ResNet. Font size changed.

- Throughout all numerical examples in this section, employing 5 hidden layers consistently produced favorable outcomes.

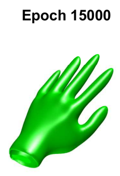
These findings emphasize the delicate balance between the number of hidden layers and the overall quality of the generated surfaces. The results underscore the importance of selecting an appropriate number of layers.



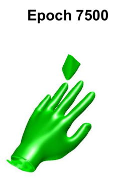
(a) $n_l = 3$



(b) $n_l = 5$



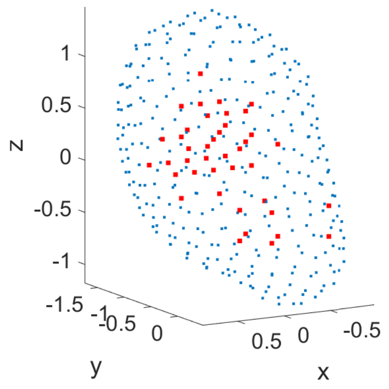
(c) $n_l = 10$



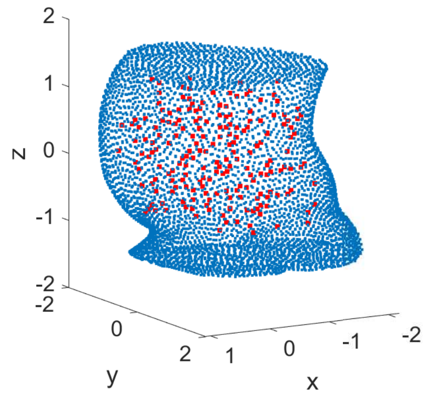
(d) $n_l = 15$

Figure 8: The profile of the simulated hand. The top panel: Plain NN, middle panel: ResNet, bottom panel: SQR-ResNet. Font size changed.

Example 3 In this instance, we embark on the endeavor of reconstructing the three-dimensional surface of the pregnant human uterus and abdomen where the point distribution are represented in Figure 9(a) and Figure 9(b), respectively. The precise reconstruction of the uterus-abdomen geometry carries significant importance, particularly in maintaining the accuracy of electro-myometrial imaging (EMMI). EMMI is an innovative electro-physiology imaging modality designed to non-invasively capture the electrical activation patterns of the uterus as it undergoes mechanical contraction.



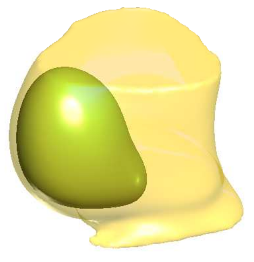
(a)



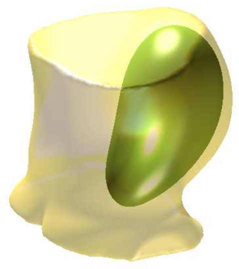
(b)

Figure 9: The profile of the point distribution in human (a) abdomen and (b) uterus.

To gather the data points for the abdomen, we employed an optical scanning device. For the uterus, data points were obtained through Magnetic Resonance Imaging (MRI). It is worth noting that MRI is a costly procedure and can be conducted exclusively in clinics equipped with MRI facilities. All data were graciously provided by the Integrated Biomedical Imaging Laboratory at the Department of Obstetrics and Gynecology, School of Medicine, Washington University in St. Louis.



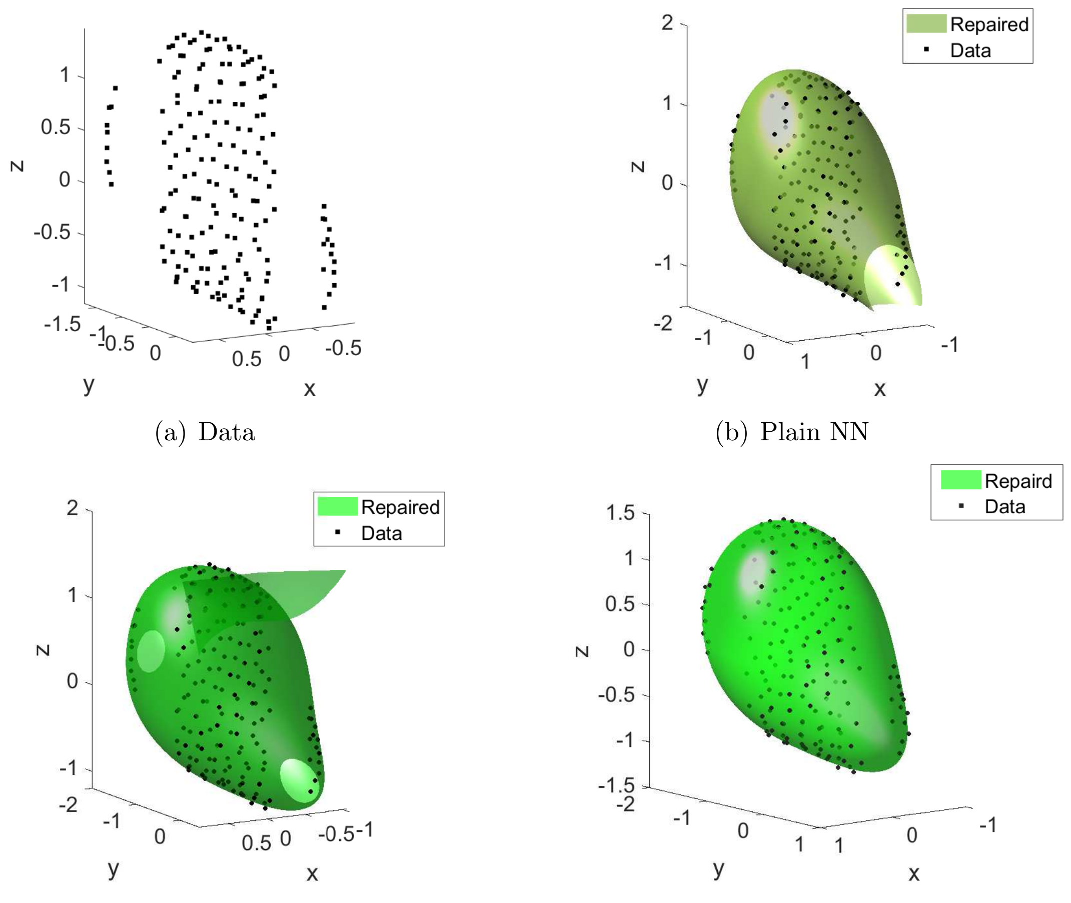
(a)



(b)

Figure 10: The profile of the simulated body and uterus.

For this example, we utilized 5000 data points for reconstructing the abdominal surface and 300 points for the uterus. The network consists of five hidden layers, each



(a) Data (b) Plain NN

(c) ResNet (d) SQR-ResNet

Figure 11: The profile of the (a) point distribution on the surface of uterus with missing data (b) simulated one using SQR-ResNet, and (c) a comparison between the full data with missing data. It looks more clear now.

with 50 neurons. The final results, depicted in Fig. 10, showcase impressively accurate reconstructions from various perspectives.

We also explored a scenario involving missing data points, specifically in the context of uterine surface reconstruction, as illustrated in Figure 11. This scenario entailed two distinct areas of missing data: one in the upper region and another in the lower part. In Figure 11(a), the available data points are highlighted, totaling 182 points, while it's noteworthy that 40

Subsequently, Figure 11(b) reveals that the algorithm effectively predicted the surface in the upper region with missing data. However, in the lower region where data was absent, the algorithm struggled to reconstruct the surface. Furthermore, Figure 11(c) demonstrates the uterine surface reconstruction using the ResNet network. Similar to the Plain NN results, a non-contiguous surface is observed in the region with missing data in the lower part, along with the appearance of an additional surface.

On the other hand, the utilization of the SQR-ResNet yielded remarkable results, effectively predicting the surface over the missing data points, highlighting its robust capabilities in handling such challenges.

Example 4 In our final example, we tackle the reconstruction of a complex domain, the Stanford Bunny [18] as the point distribution is shown in Figure 12.

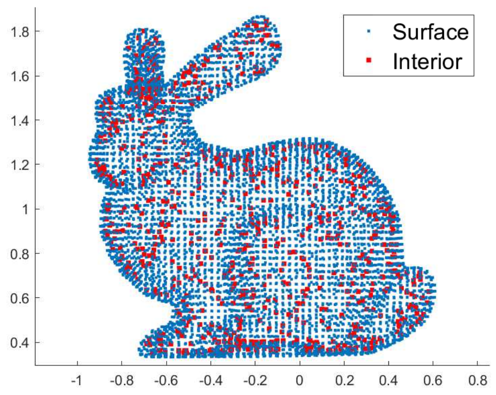


Figure 12: The profile of the point distribution for the Stanford bunny.

Figure 13 provides the outcomes when applying the Plain NN (left panel), ResNet (middle panel), and the SQR-ResNet (right panel). The top row showcases the bunny's front view, while the bottom row reveals the rear view. For this illustration, we maintain a consistent set of parameters, $n_s = 8000$ and $n_i = 3000$. We also implemented a neural network with five hidden layers, each containing 50 neurons. Notably, the Plain NN results in a relatively unclear representation of the bunny, particularly when viewed from the rear perspective (see Fig. 13(d)). In contrast, both ResNet and SQR-ResNet deliver more accurate representations. However, as previously observed, the utilization of ResNet introduces some additional surface features.

To further enhance our results, we decide to introduce 1000 exterior points (n_e) and re-run the simulation using both Plain NN and ResNet. The outcomes are presented in Figure 14, with the left panel depicting the Plain NN results and the right panel showing the results achieved using the ResNet approach. As evident, the results obtained using Plain NN exhibit a significant improvement compared to those shown in Figure 13(a) and Figure 13(d). Moreover, the ResNet results also display a significant refinement in surface construction when exterior points are introduced. This underlines the substantial impact of exterior points on the quality of reconstruction.

5 Conclusion

In this study, we have explored the application of advanced neural network architectures, including Plain Neural Networks, Residual Networks (ResNet), and a novel variant known

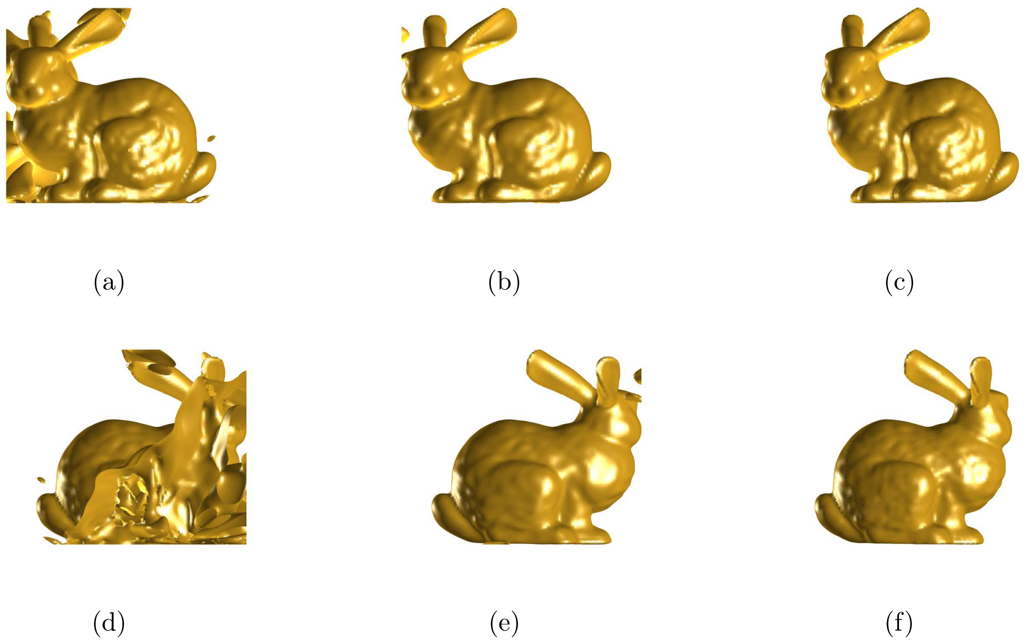


Figure 13: The surface of the simulated Stanford bunny. The left panel: Plain NN, middle panel: ResNet, and right panel: SQR-ResNet.

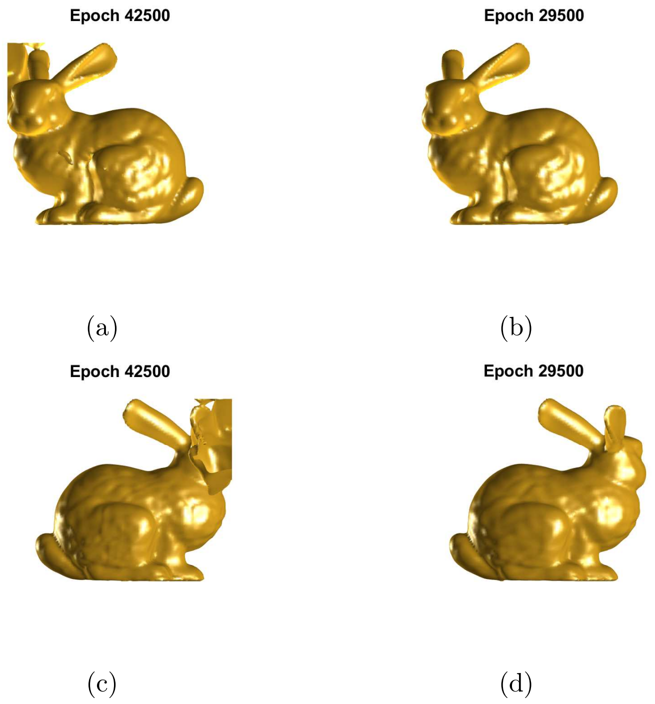


Figure 14: The enhanced surface of the simulated Stanford bunny using the exterior points. The left panel: Plain NN, right panel: ResNet, with $n_i = 1000$, $n_e = 1000$

as SQR-ResNet, for the reconstruction of surfaces from data points in various scenarios.

The results from our investigations demonstrate the potential of these architectures in surface reconstruction applications.

Our findings indicate that the SQR-ResNet architecture, with its unique inclusion of power-enhanced layers, offers significant advantages. This architecture consistently exhibits superior quality of surface reconstruction making it a valuable tool for a wide range of applications.

Through a series of numerical examples, we have observed that the number of interior points, the distribution of data, and the complexity of the domain all play crucial roles in the performance of the neural network architectures. A moderate number of hidden layers $n_l = 5$ and neurons $n_n = 100$ appear to be optimal for achieving smoother and more accurate surface representations.

Moreover, in a medical context, our study demonstrates that these advanced architectures can contribute to the accurate reconstruction of complex anatomical structures, such as the uterus during pregnancy. The SQR-ResNet excels in cases of missing data, providing the capability to predict surfaces even when data points are incomplete, which is essential for medical imaging applications.

For future studies, we are planning to incorporate physics-informed neural networks (PINN) as a promising approach to enhance the accuracy of surface reconstruction. This method leverages physics principles to refine the reconstruction process, offering potential improvements in accuracy and reliability [19].

References

[1] P. Erler, S. Ohrhallinger, N. Mitra, and M. Wimmer. "Points2Surf: Learning implicit surfaces from point clouds," in European Conference on Computer Vision (ECCV), 2020.

[2] H. Zheng, F. Wang, C. S. Chen, M. Lei and Yong Wang. Improved 3D Surface Reconstruction via the Method of Fundamental Solutions. Numerical Mathematics: Theory, Methods and Applications (4)973-985, 2020.

[3] X.-Y. Liu, H. Wang, C.S. Chen, Q. Wang, X. Zhou, and Y. Wang. Implicit surface reconstruction with radial basis functions via PDEs. Engineering Analysis with Boundary Elements, 110: p. 95-103, 2020.

[4] J. Digne, J.M. Morel, C.M. Souzani, C. Lartigue. Scale space meshing of raw data point sets. In: Computer Graphics Forum, vol. 30, pp. 1630–1642. Wiley Online Library, 2011.

[5] S. Ohrhallinger, S. Mudur, M. Wimmer. Minimizing edge length to connect sparsely sampled unstructured point sets. Comput. Graph. 37(6), 645–658, 2013.

[6] G. Li, L. Liu, H. Zheng, N.j. Mitra. Analysis, reconstruction and manipulation using arterial snakes. In: ACM SIGGRAPH Asia 2010 Papers, SIGGRAPH ASIA 2010. Association for Computing Machinery, New York, 2010.

[7] F. Williams, T. Schneider, C.T. Silva, D. Zorin, J. Bruna, D. Panozzo. Deep geometric prior for surface reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 10130–10139, 2019.

[8] M. Kazhdan, M. Bolitho, H. Hoppe. Poisson surface reconstruction. In: Proceedings of the Eurographics Symposium on Geometry Processing, 2006.

[9] M. Kazhdan, H. Hoppe. Screened Poisson surface reconstruction. ACM Trans. Graph. (ToG) 32(3), 29, 2013.

[10] T. Groueix, M. Fisher, V.G. Kim, B. Russell, M. Aubry. AtlasNet: a Papier- Mâché approach to learning 3D surface generation. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[11] A. Dai, M. Nießner. Scan2Mesh: from unstructured range scans to 3D meshes. In: Proceedings of the Computer Vision and Pattern Recognition (CVPR), IEEE, 2019.

[12] A. Noorizadegan, D.L. Young, Y.C. Hon, C.S. Chen. Physics-Informed with Power-Enhanced Residual Network for Interpolation and Inverse Problems. arXiv preprint, arXiv:2310.15690, 2023.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. arXiv preprint arXiv:1603.05027, 2016.

[15] H. Li, Z. Xu, G. Taylor, Christoph Studer, Tom Goldstein: Visualizing the Loss Landscape of Neural Nets, arxiv.1712.09913v3, 2018.

[16] S. Jastrzębski, Arpit D, Ballas N, Verma V, Che T, Bengio Y: Residual Connections Encourage Iterative Inference. arXiv:1710.04773, 2017.

[17] A. Veit, M. Wilber, and S. Belongie, Residual networks behave like ensembles of relatively shallow networks, in Proceedings of the 30th International Conference on Neural Information Processing Systems. Curran Associates Inc.: Barcelona, Spain. p. 550–558, 2016.

[18] The Stanford 3D Scanning Repository, Stanford Computer Graphics Laboratory, http://graphics.stanford.edu/data/3Dscanrep/.

[19] M. Raissi, P. Perdikaris, and G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378: p. 686-707, 2019.