Accelerating AI Performance using Anderson Extrapolation on GPUs

Saleem Abdul Fattah Ahmed Al Dajani*, **David E. Keyes**
Applied Physics Program, Physical Science and Engineering Division
Extreme Computing Research Center, Computer, Electrical, Mathematical, Science and Engineering Division
King Abdullah University of Science and Technology (KAUST)
Thuwal, Makkah Province, Kingdom of Saudi Arabia (KSA) 23955-6900
saleem.aldajani@kaust.edu.sa, david.keyes@kaust.edu.sa

Abstract

We present a novel approach for accelerating AI performance by leveraging Anderson extrapolation, a vector-to-vector mapping technique based on a window of historical iterations. By identifying the crossover point where a mixing penalty is incurred, the method focuses on reducing iterations to convergence, with fewer more compute-intensive but generally cacheable iterations, balancing speed and memory usage with accuracy and algorithmic stability, respectively. We demonstrate significant improvements, in both training and inference, motivated by scalability and efficiency extensions to the realm of high-performance computing (HPC).

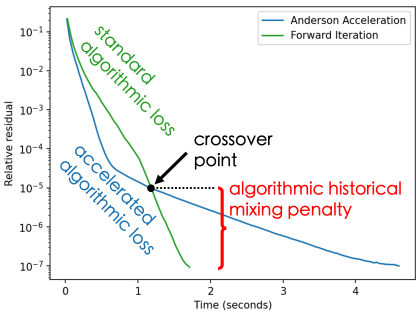


Figure 1: Crossover and mixing penalty plotted against time. Relative residual is $\frac{\|f(z^k,x)-z^k\|_2}{\|f(z^k,x)\|_2+\lambda}$ [34].

1 Introduction

Anderson extrapolation [1, 2, 16, 27, 33] has recently been applied to deep equilibrium models (DEQs) [7–10, 17, 24]. Kolter et al. [34] found the gains not substantial due to early termination with a loose convergence tolerance. They focused on Anderson extrapolation during training. Here, we show significant acceleration of AI performance with Anderson on GPUs for both the forward pass (running inferences faster) and training (generating models faster). We demonstrate acceleration of the forward pass with standard Anderson as a baseline for future work with stochastic variants [30] and accelerating the backward pass with Jacobian-free methods like Jacobian-Free Backpropagation (JFB) and Neumann series gradient approximations [16].

As AI demand grows, as shown in Fig. 2 [3, 15, 25, 28], high-performance computing (HPC) is becoming critical due to economic pressures from the growth of data and AI infrastructure [29]. Low-memory acceleration techniques, like Anderson extrapolation, will be key to increasing HPC-based AI computational efficiency. This study investigates matrix-free Anderson extrapolation on GPUs, emphasizing gains from advanced computing architectures compared to CPUs. Our goal is to maximize computational efficiency while reducing iterations to convergence by reusing previous

*Code can be found at http://tinyurl.com/DeepAndersoNN

38th Second Workshop on Machine Learning with New Compute Paradigms at NeurIPS 2024(MLNCP 2024).

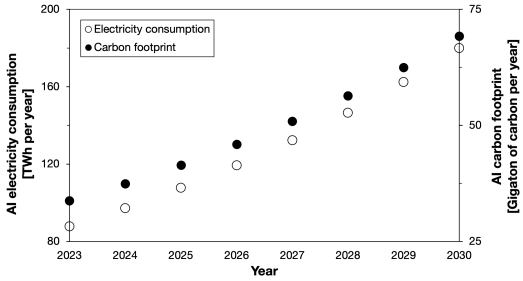


Figure 2: AI carbon footprint projected to consume >2% of global electricity demand [3, 15, 25, 28], amounting to >10% of global electricity demand for data centers and infrastructure.

iterations to avoid unnecessary gradient calculations, gaining benefits expected from second-order methods (e.g., [32]) without manipulating Hessian matrices.

The environmental impact of AI is rapidly growing [3, 15, 25, 28]. By 2030, AI is projected to account for 2% of global electricity consumption. We aim to reduce this impact by up to 90%, saving 160 terawatt-hours per year by 2030. The carbon footprint of AI exceeds the 500-megaton annual benchmark set by initiatives like Bill Gates' Breakthrough Energy [14]. Efficiency-enhancing technologies like GPU and Anderson acceleration can reduce AI's carbon emissions by 60 gigatons per year by 2030, as shown in Fig. 2.

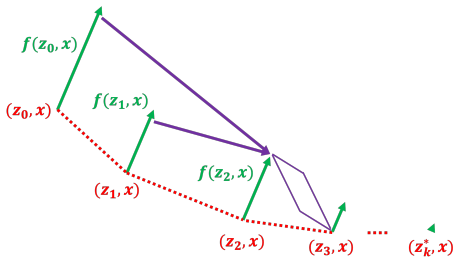
1.1 Leveraging extrapolation for AI and HPC advances

Anderson extrapolation, a windowing technique for accelerating nonlinear fixed point iterations, is widely applied in fields like density functional theory, kinetic theory, and climate spin-up. It is well-suited for distributed memory parallelization and GPU implementation. It is a staple of major open-source large-scale solver libraries, including PETSc [11, 12], SUNDIALS [23], Trilinos [19–22], and deal.II [4–6, 13]. It can be applied to machine learning training, smoothing out standard forward iterations and achieving superior accuracy in training and testing error. Benchmarking results on CIFAR10 show expected robustness benefits and allow characterization of the temporal advantages or disadvantages from the higher cost per iteration, where a small residual minimization step is applied at each new function evaluation.

$$z_{k+1} = f(z_k, x)\,, z_k \in \mathbb{R}^n\,, k = 0, 1, 2, \ldots$$

$$r_k(x_k) = z_k - z(x_k), w_i^{(k)} = \frac{-r_{i,k}^T(r_{i,k} - r_{i,k-1})}{\left\|r_{i,k} - r_{i,k-1}\right\|^2}$$

$$z_{k+1} = f(z_k, x) + \sum_{i=1}^{min(k,m)} w_i^{(k)}\big(f(z_k, x) - f(z_{k-1}, x)\big)$$



$$z_3 = f(z_2) + w_1^{(2)}\big(f(z_2, x) - f(z_1, x)\big) + w_2^{(2)}\big(f(z_2, x) - f(z_0, x)\big)$$

Figure 3: Mathematical formulation and vector representation. Adapted from Y. He & H. De Sterck. "Linear Asymptotic Convergence Analysis of Anderson Acceleration, with Krylov Formulation in the Linear Case" Copper Mountain Conference (2022), ICERM Workshop (2023). Available at: `https://www.bilibili.com/video/BV1Wa411i77y/` and `https://icerm.brown.edu/video_archive/?play=3320`

1.2 Balancing memory and stability

Fundamental tradeoffs exist between memory capacity, memory bandwidth, communication cost, and algorithmic characteristics of stability and convergence rate. The tradeoffs are generally resolved

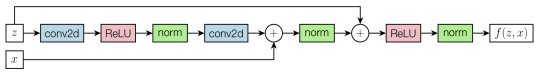


Figure 4: Deep equilibrium neural network model architecture (Source: NeurIPS Tutorial, 2020 [34]). $f(z, x) = \mathrm{norm}(\mathrm{ReLU}(z + \mathrm{norm}(x + W_2 * (\mathrm{norm}(\mathrm{ReLU}(W_1 * z))))))$ "norm" here is a group norm, representing a statistical normalization [31].

to minimize time to solution. GPUs attain high memory bandwidth advantages over CPUs at the cost of smaller memory capacity. Anderson extrapolation promotes fewer, more expensive steps, reusing cached state-vector data. In distributed memory implementations, it produces convergence with fewer interprocessor communication steps. It has tuning parameters such as window size and damping that can be tuned to application and architecture. We are assessing its utility in machine learning more broadly at a time of emergent CPU-GPU superchips.

1.3 Deep equilibrium neural network models

Deep equilibrium models (DEQs) are the continuum limit of explicit neural networks as the number of layers approaches infinity [26], approximating many explicit layers with a single, implicit layer with exponentially fewer parameters using a backward pass including the output. This reduces the inverse problem in parameter space to a fixed-point iteration problem, enabling the usage of nonlinear, vector-to-vector mapping techniques to compute the fixed-point iterations that converge to the deep equilibrium state parameters by minimizing the loss function. With gains in memory and acceleration, DEQs are fit for large-scale computer vision and natural language processing tasks and benefit more from matrix-vector operation-optimized computing architectures like GPUs and CPU-GPU superchips.

The standard approach using forward iteration for fixed-point iteration problems often does not efficiently converge to the fixed point and suffers from initially slow error reduction and local minimum trapping in nonlinear problems like deep neural networks. Vector-to-vector mapping techniques like Anderson extrapolation outperform standard forward iteration by combining information from previous iterations to span a searchable subspace to extrapolate the next iteration, enhancing convergence rates at the expense of memory usage in each iteration.

DEQs represent any neural network at arbitrary depths and connectivities with a single implicit layer consuming vastly fewer parameters with faster forward passes for accelerated training and inferences. The implicit function theorem shows how gradients can be computed in the DEQ framework, facilitating backpropagation through the equilibrium state [8, 34].

DEQs provide a framework for accelerating deep learning, extending the capacity of deep networks within a single-layer architecture through fixed-point computations and advanced root-finding algorithms. Their amenability to convergence acceleration with techniques like Anderson positions DEQs as a robust method to reduce computation needed to build state-of-the-art models and scale up beyond current computational limitations.

2 Methods

This work demonstrates Anderson extrapolation to accelerate AI performance algorithmically without increasing processors. Since it is a vector-to-vector mapping technique, it benefits from hardware optimized for uniform operations, like GPUs. We benchmark Anderson acceleration against standard forward iteration on GPUs and CPUs.

2.1 Mathematical formulation

Fixed-point acceleration starts with the fixed point iteration formula $z^\star = f(z^\star, x)$. Forward iteration, $z^{k+1} = f(z^k, x)$, moves step-wise towards this fixed point.

Anderson acceleration uses a linear combination of prior iterates, $z^{k+1} = \sum_{i=1}^{m} \alpha_i f(z^{k-i+1}, x)$, optimizing α_i to minimize the residual norm, $\frac{\|f(z^k,x)-z^k\|_2}{\|f(z^k,x)\|_2+\lambda}$, leading to faster convergence. The coefficients must sum to unity:

$$\mathrm{minimize}_\alpha \quad \|G\alpha\|_2^2, \quad \text{subject to} \quad 1^T\alpha = 1 \tag{1}$$

The matrix G is defined as:

$$G = \left[f(z^k, x) - z^k, \cdots, f(z^{k-m+1}, x) - z^{k-m+1}\right] \tag{2}$$

The Lagrangian incorporating the equality constraint is:

$$L(\alpha, \nu) = \|G\alpha\|_2^2 - \nu(1^T\alpha - 1) \tag{3}$$

Algorithm 1 Extrapolation for Fixed Point Iteration [34]

Input: Function f, initial guess x_0, window size $m = 5$, regularization $\lambda = 1e-5$, max iterations
$max_iter = 1000$, tolerance $tol = 1e-2$, mixing parameter $\beta = 1.0$
Initialize n, batch size b, channels d, height H, width W from $x_0.shape$
$X, F \leftarrow$ Initialize tensors based on b, m, and $d \times H \times W$
$H, y \leftarrow$ Initialize for least squares solver
$times, res \leftarrow$ Initialize lists for timing and residuals
for $k = 2$ **to** max_iter **do**
  Start timing this iteration
  $n \leftarrow \min(k, m)$
  $G \leftarrow F[:, :n] - X[:, :n]$
  Update H using G
  Solve linear system for α
  Update X and F using α, m, and β
  Compute residual, $\frac{\|f(z^k,x)-z^k\|_2}{\|f(z^k,x)\|_2+\lambda}$
  Store time and residual
  Check for convergence
  if residual $<$ tol **then**
    break
  end if
end for
return $X[:, k\%m](x_0)$, residuals, times

To solve for α_i, we set up and solve:

$$\begin{bmatrix} 0 & 1^T \\ 1 & H \end{bmatrix} \vec{y} = \begin{bmatrix} 0 & 1^T \\ 1 & G^T G + \lambda I \end{bmatrix} \begin{bmatrix} \nu \\ \alpha \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{4}$$

Anderson acceleration can incorporate a mixing parameter β:

$$z^{k+1} = (1-\beta) \sum_{i=1}^{m} \alpha_i z^{k-1+1} + \beta \sum_{i=1}^{m} \alpha_i f(z^{k-i+1}, x) \tag{5}$$

2.2 Dataset description, compute environment, and training details

The CIFAR10 dataset, with 60,000 32x32 labeled images in 10 classes, is used for supervised learning and image classification tasks. Accuracy is the ratio of correctly predicted labels to the total images, using cross-entropy loss.

High-dimensional tensors in standard PyTorch format are used. The compute environment includes Google Colab Pro with NVIDIA Tesla V100 GPUs and Intel Xeon CPUs. Training uses default hyperparameters from Kolter et al. [34] for comparison with prior results [7–10, 17, 24], with Anderson parameters $m = 5$ and $\beta = 1$.

2.3 Deep neural networks, deep equilibrium models, and fixed Point equations

Traditional neural networks use layer-wise transformations:

$$\begin{aligned} z_1 &= x \\ z_{i+1} &= \sigma(W_i z_i + b_i), \quad i = 1, \ldots, k-1 \\ h(x) &= W_k z_k + b_k \end{aligned}$$

DEQs model a network as an infinitely deep system, finding a fixed point $z^\star$ that satisfies:

$$z^\star = \sigma(W z^\star + U x + b) \tag{6}$$

Here, W, U, and b are shared across all layers, and σ is the activation function. Solving for $z^\star$ avoids computing individual layers, reducing computational cost.

2.4 GPU Optimization and Parallelization

GPUs, suited for uniform tasks with high throughput, map well with Anderson acceleration. This work combines suitable algorithms with appropriate architecture to enhance performance without upgrading hardware or using more processors.

3 Results

This work demonstrates that Anderson extrapolation has a higher cost per iteration, measured in function evaluations or epochs. The main benefit is that Anderson extrapolation exhibits less fluctuation in accuracy, as seen in the test accuracy, whereas forward iteration shows more significant ups and downs in both training and testing accuracy, potentially indicating overfitting. Anderson acceleration reaches a higher accuracy plateau for both training and test datasets, suggesting better generalization capability.

Table 1: Summary of algorithmic improvements to training and inference without augmentation.

	Algorithm	**DEQ (ours)**	DEQ [Implicit] [8]	ResNet-18 **[Explicit]** [18]
Number of parameters	Standard	**64,842**	∼170,000	∼170,000
	Accelerated	**64,842**	-	-
Training accuracy	Standard	64.7%	-	-
	Accelerated	**96.3%**	-	-
Testing accuracy	Standard	64.2%	82.2%	81.6%
	Accelerated	**79.1%**	-	-
Training time [seconds]	Standard	1.2×10^4	-	-
	Accelerated	$\mathbf{1.4\times10^3}$	-	-
Inference time [seconds]	Standard	1	-	-
	Accelerated	**0.5**	-	-
Speedup relative to standard	Ratio	**2-8.6**	-	-
	Compute saved	**50-88%**	-	-

Anderson extrapolation is benchmarked against traditional forward iteration methods in DEQs to understand its role in AI and HPC. The computational demand of Anderson extrapolation correlates with the number of epochs, as shown in Fig. 7. A trade-off is shown between accuracy and computing time, whereas forward iteration maintains a more consistent computational time as the number of epochs increases.

Implicit neural network model architecture performance is analyzed with different vector-to-vector mapping techniques. The goal is to understand how incorporating Anderson acceleration impacts model accuracy and performance. The stability of train and test accuracy is observed, and Anderson acceleration demonstrates higher consistency over numerous epochs, whereas forward iteration reveals significant swings in train and test accuracy. Initialization error with Anderson is lower than with forward iteration.

Anderson acceleration reaches higher accuracies in training and testing in less time than forward iteration. Anderson acceleration is also superior with random inputs. Across testing with random inputs, Anderson acceleration consistently outperforms or matches forward iteration, depending on target relative residual accuracy.

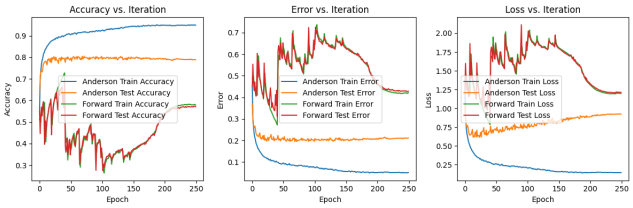


Figure 5: Evaluating CIFAR10 dataset through deep equilibrium. Anderson is 1.2x more accurate at stable convergence above mixing penalty.

4 Discussion

These results show that Anderson extrapolation can train DEQ networks to higher accuracy than forward iterations and reach a given high accuracy in less time. Anderson extrapolation is also efficiently implementable in GPU programming environments, utilizing memory austerity and operational uniformity attributes similar to the forward algorithm. For large-scale neural network training problems requiring distributed memory, this study motivates porting and testing on state-of-the-art GPU architectures, CPU-GPU superchips, and emerging computing hardware.

GPUs have been shown to accelerate Anderson extrapolation beyond what could be achieved with standard forward iterations or with Anderson on CPUs. This is notable before reaching the 'crossover point,' the trade-off between computation speed and accuracy, illustrated in Fig. 1 and Fig. 6. The 'mixing penalty' due to the additional computational cost associated with Anderson acceleration is offset by the parallel processing capabilities of GPUs, enabling faster convergence than with CPUs or standard forward iterations alone.

The increase in time per iteration with Anderson arises from the residual minimization process during each acceleration step. The higher plateau for accuracy with Anderson compared to forward iteration suggests more robust learning when taking previous iterations into account. Monitoring the slowing of Anderson acceleration and switching to approximate forms of Newton's method (e.g., quasi-Newton, modified Newton, or inexact Newton) can be beneficial.

The unstable behavior with forward iteration necessitates lower learning rates and more epochs for training, increasing the time needed to reach the same accuracies achieved with Anderson by up

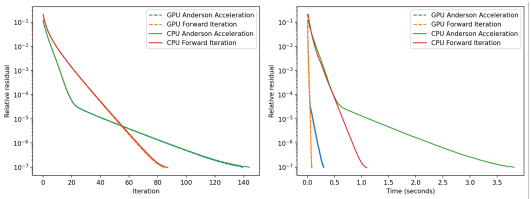


Figure 6: Evaluating relative residual, $\frac{\|f(z^k,x)-z^k\|_2}{\|f(z^k,x)\|_2+\lambda}$, for a random input x. A typical GPU is $\sim$100-150x faster to target relative residual than a typical CPU using Anderson, with a mixing penalty that is $\sim 10^{-1} - 10^{-2}$ lower.

to an order of magnitude. The inconsistency in accuracy with forward iteration raises concerns about overfitting during training, undermining the model's ability to generalize for reliable predictions on new, unseen data.

These findings indicate that Anderson acceleration improves DEQ performance with more rapid error reduction at the outset, as shown in Fig. 6 and Fig. 7. The rate at which peak accuracy is reached with extrapolation enables peak neural network performance in a fraction of the time required for forward iterations to stabilize at comparable accuracy. This acceleration is beneficial in time-sensitive applications where rapid deployment of accurate AI models is essential.

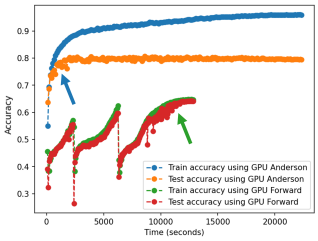


Figure 7: Deep equilibrium model is $\sim$10x faster to stable convergence with Anderson relative to standard forward iteration, per Tab. 1

5 Conclusion

The integration of Anderson acceleration within deep learning workflows presents substantial improvements in computational efficiency, accuracy, and generalizability of implicit neural networks. Porting and parallelizing vector-to-vector mapping techniques onto emerging CPU-GPU hybrid architectures holds promise. The accuracy and speed of deep equilibrium neural network training and inferences could be improved further, making them more viable for real-world applications beyond the classification task demonstrated herein. Based on investigations of explicit and implicit memory requirements [26], optimizations based on an Anderson-accelerated, fixed-point iteration implicit memory approach [34] are effective in memory-intensive computer vision processing, reducing memory and bandwidth consumption without compromising performance [26].

These methods applied to implicit neural networks, particularly DEQs, reveal new directions for AI research, such as exploring further acceleration gains from stochastic variants of Anderson extrapolation [30]. Exploiting the continuum limit of infinite explicit layers in implicit networks reduces memory usage and achieves favorable performance trade-offs [8], where gradient approximations, such as truncated backward gradient for backpropagation [16, 24], can be applied for even more acceleration.

6 NeurIPS Limitation and Broader Impact Statements

These results do not comprehensively search the Anderson hyperparameter space, nor do they establish the multiprocessor scalability at which they are aimed. Saving training and inference time and energy is the broader impact envisioned for this work. Being algorithmic in nature, it has the same potential for applied use and misuse as neural networks in general.

Acknowledgments and Disclosure of Funding

SAA acknowledges funding from the KAUST Fellowship, the ECRC, and Prof. David Keyes throughout the duration of this work. SAA wishes to express sincere gratitude to Henning Soller, Mohamed-Slim Alouni, Matteo Parsani, George Turkiyyah, and Frederic Laquai for their invaluable chats throughout this research. Special gratitude is extended to my family—late grandparents, parents, wife, daughter, and brothers—and to my friends for their unwavering support and encouragement throughout the duration of this study.

References

[1] Donald G Anderson. Iterative Procedures for Nonlinear Integral Equations. *Journal of the Association for Computing Machinery (JACM)*, 12(4):547–560, 1965.

[2] Donald GM Anderson. Comments on "Anderson Acceleration, Mixing and Extrapolation". *Numerical Algorithms*, 80:135–234, 2019.

[3] Anders S.G. Andrae and Tomas Edler. On Global Electricity Usage of Communication Technology: Trends to 2030. *Challenges*, 6(1):117–157, 2015.

[4] Daniel Arndt, Wolfgang Bangerth, Denis Davydov, Timo Heister, Luca Heltai, Martin Kronbichler, Matthias Maier, Jean-Paul Pelteret, Bruno Turcksin, and David Wells. The DEAL.II finite element library: Design, features, and insights. *Computers & Mathematics with Applications*, 81:407–422, 2021.

[5] Daniel Arndt, Wolfgang Bangerth, Marco Feder, Marc Fehling, Rene Gassmöller, Timo Heister, Luca Heltai, Martin Kronbichler, Matthias Maier, Peter Munch, et al. The deal.II Library, Version 9.4. *Journal of Numerical Mathematics*, 30(3):231–246, 2022.

[6] Daniel Arndt, Wolfgang Bangerth, Maximilian Bergbauer, Marco Feder, Marc Fehling, Johannes Heinz, Timo Heister, Luca Heltai, Martin Kronbichler, Matthias Maier, et al. The deal. II Library, Version 9.5. *Journal of Numerical Mathematics*, 31(3):231–246, 2023.

[7] Shaojie Bai. *Equilibrium Approaches to Modern Deep Learning*. PhD thesis, Carnegie Mellon University, 2022.

[8] Bai, Shaojie and Kolter, J Zico and Koltun, Vladlen. Deep Equilibrium Models. *Advances in Neural Information Processing Systems*, 32, 2019.

[9] Bai, Shaojie and Koltun, Vladlen and Kolter, J Zico. Multiscale Deep Equilibrium Models. *Advances in Neural Information Processing Systems*.

[10] Bai, Shaojie and Koltun, Vladlen and Kolter, J Zico. Stabilizing Equilibrium Models by Jacobian Regularization. *arXiv:2106.14342*, 2021.

[11] Satish Balay, Kris Buschelman, William D Gropp, Dinesh Kaushik, Matthew G Knepley, L Curfman McInnes, Barry F Smith, and Hong Zhang. PETSc. *See http://www.mcsanl.gov/petsc*, 2001.

[12] Satish Balay, Shrirang Abhyankar, Mark Adams, Jed Brown, Peter Brune, Kris Buschelman, Lisandro Dalcin, Alp Dener, Victor Eijkhout, William Gropp, et al. PETSc Users Manual. 2019.

[13] Wolfgang Bangerth, Ralf Hartmann, and Guido Kanschat. deal.II — A General-Purpose Object-Oriented Finite Element Library. *ACM Transactions on Mathematical Software (TOMS)*, 33(4): 24–es, 2007.

[14] Catherine Clifford. Bill Gates: These 5 concepts will help you understand the urgency of the climate crisis, 2021. URL https://www.cnbc.com/2021/02/14/bill-gates-concepts-to-understand-the-climate-crisis.html.

[15] Alex de Vries. The Growing Energy Footprint of Artificial Intelligence. *Joule*, 7(10):2191–2194, 2023.

[16] Fung, Samy Wu et al. JFB: Jacobian-Free Backpropagation for Implicit Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6648–6656, 2022.

[17] Geng, Zhengyang and Zhang, Xin-Yu and Bai, Shaojie and Wang, Yisen and Lin, Zhouchen. On Training Implicit Models. *Advances in Neural Information Processing Systems*, 34:24247–24260, 2021.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[19] Michael Heroux, Roscoe Bartlett, Vicki Howle Robert Hoekstra, Jonathan Hu, Tamara Kolda, Richard Lehoucq, Kevin Long, Roger Pawlowski, Eric Phipps, Andrew Salinger, et al. An Overview of Trilinos. 2003.

[20] Michael A Heroux and James M Willenbring. A new overview of the trilinos project. *Scientific Programming*, 20(2):83–88, 2012.

[21] Michael A Heroux, Roscoe A Bartlett, Vicki E Howle, Robert J Hoekstra, Jonathan J Hu, Tamara G Kolda, Richard B Lehoucq, Kevin R Long, Roger P Pawlowski, Eric T Phipps, et al. An overview of the trilinos project. *ACM Transactions on Mathematical Software (TOMS)*, 31 (3):397–423, 2005.

[22] Michael Allen Heroux and James M Willenbring. Trilinos Users Guide. 2003.

[23] Alan C Hindmarsh, Peter N Brown, Keith E Grant, Steven L Lee, Radu Serban, Dan E Shumaker, and Carol S Woodward. SUNDIALS: Suite of Nonlinear and Differential/Algebraic Equation Solvers. *ACM Transactions on Mathematical Software (TOMS)*, 31(3):363–396, 2005.

[24] Zhichun Huang, Shaojie Bai, and J Zico Kolter. $(\text{Implicit})^2$: Implicit Layers for Implicit Representations. *Advances in Neural Information Processing Systems*, 34:9639–9650, 2021.

[25] Nicola Jones et al. How to Stop Data Centres from Gobbling Up the World's Electricity. *Nature*, 561(7722):163–166, 2018.

[26] Guohao Li, Matthias Müller, Bernard Ghanem, and Vladlen Koltun. Training Graph Neural Networks with 1000 Layers. In *International Conference on Machine Learning*, pages 6437–6449. PMLR, 2021.

[27] Ouyang, Wenqing et al. Anderson Acceleration for Nonconvex ADMM Based on Douglas-Rachford Splitting. In *Computer Graphics Forum*, volume 39, pages 221–239. Wiley Online Library, 2020.

[28] David Patterson et al. Carbon Emissions and Large Neural Network Training. *arXiv: 2104.10350*, 2021.

[29] Schwarz, Nicholas et al. Enabling Scientific Discovery at Next-Generation Light Sources with Advanced AI and HPC. In *Driving Scientific and Engineering Discoveries Through the Convergence of HPC, Big Data and AI: 17th Smoky Mountains Computational Sciences and Engineering Conference, SMC 2020, Oak Ridge, TN, USA, August 26-28, 2020, Revised Selected Papers 17*, pages 145–156. Springer, 2020.

[30] Fuchao Wei, Chenglong Bao, and Yang Liu. Stochastic Anderson Mixing for Nonconvex Stochastic Optimization. *Advances in Neural Information Processing Systems*, 34:22995–23008, 2021.

[31] Yuxin Wu and Kaiming He. Group Normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[32] Stefano Zampini, Umberto Zerbinati, George Turkiyyah, and David Keyes. PETScML: Second-order solvers for training regression problems in Scientific Machine Learning. *arXiv:2403.12188*, 2024.

[33] Guang-Yu Zhu, Wei-Dong Li, and Wei Hong. An Integral Equation-Based Overlapping Domain Decomposition Method Enhanced With Anderson Acceleration. *IEEE Antennas and Wireless Propagation Letters*, 22(3):492–496, 2022.

[34] Zico Kolter, David Duvenaud, and Matt Johnson. Deep Implicit Layers - Neural ODEs, Deep Equilibirum Models, and Beyond. NeurIPS Tutorial. 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims are mentioned in Section 1.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [No]

Justification: Theoretical assumptions and proofs are outside the scope and length restrictions of this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Information to reproduce the main experimental results of the paper are included in the text or references.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Data and code used in the paper are openly accessible and reported in Section 2.2.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Necessary training and test details are included in Section 2.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: Experimental statistical significance and error bars are not applicable to the results reported in this paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computer resources needed to reproduce the experiments are included in Section 2.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Research conducted in this paper conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts are described in Section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Safeguards are not applicable to any release in this paper.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Data and code used in this paper are properly described and cited in Section 2.2.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new assets are introduced in this paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No human subjects associated with this paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects are associated with this paper, so no IRB approvals are applicable.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.