Reproducibility and Ablation Study of "Augmented Neural ODEs"

Anonymous Author(s)
Affiliation
Address
email

Abstract

"Augmented Neural ODEs" (ANODE) by Dupont *et* al. addresses a central aspect of treating a neural network as an ordinary differential equation (ODE): solution trajectories of such functions may not overlap. The authors address this issue by augmenting the feature space, simplying the computation of solution trajectories. Here, we report on the reproducibility of the results presented in Dupont *et al.*, and perform ablation and robustness experiments. Most results presented in the original study are reproducible given the author's implementation. Small variations in hyper-parameters did not cause drastic changes to model the performance. We also provide additional theoretical support to their framework by measuring ANODE performance with fixed stepsize solvers. In a 2D toy dataset, the model performance improves with increased augmentation using fixed stepsize, supporting the conjecture that ANODEs simplify flows. On the MNIST dataset, ANODEs become unstable using fixed stepsize, underlining the need for adaptive stepsize methods when ANODEs are trained on large data-sets.

1 Introduction

In a 2015 survey, Nature reported that over 70% of researchers have been unable to reproduce published results and 50% failed to reproduce their own studies [1]. In a follow-up paper, Huston (2018) [3] postulated that the reproducibility crisis faced by the Machine Learning community is in part due to code being unavailable, lack of space to display hyper-parameter search and pressure to publish quickly, reducing the quality of the publication and the extent to which results are investigated. As such, reproducibility challenges such as the one hosted by NeurIPS 2019 provide an excellent opportunity to mitigate some of the concerns raised by Huston (2018).

Dupont *et al.* 2019 [5] proposed an extension to Chen *et al.*'s (2018) Neural Ordinary Differential Equations (NODEs). NODEs can be seen as continuous approximations of Residual Networks (ResNets) and can be numerically approximated using traditional Ordinary Differential Equation (ODE) solvers. As noted by Dupont *et al.*, NODEs have an important limitation due to their ODE-like nature: they cannot represent functions whose solution trajectories intersect. To overcome this limitation, the authors propose to augment the number of feature space dimensions in a method they called Augmented Neural Ordinary Differential Equations (ANODEs). With increased dimensionality, solution trajectories can be simplified and will not overlap, allowing for faster and more generalizable estimations. In their paper, Dupont *et al.* evaluate the performance and complexity of ResNets, NODEs and ANODEs on toy datasets and in several image datasets.

As part of the 2019 NeurIPS Reproducibility Challenge, we conducted a reproducibility and ablation study of Dupont *et al.* (2019). The authors provided the code used on a GitHub repository[1], which

[1]https://github.com/EmilienDupont/augmented-neural-odes

we used to replicate the main results of the paper and to test their robustness. We first provide a brief background on the problem (Section 2), report our findings (Section 3), and conclude by discussing ANODE characteristics and robustness (Section 5). Our modifications to Dupont *et al.*'s ANODE implementation and associated documentation can be found on our GitHub repository[2].

2 Background

ResNets comprise a family of deep neural networks which implement skip connections between network layers [2]. As such, ResNets send information of a hidden state at layer t to the hidden state output at layer $t + \Delta t$ where Δt specifies the number of layers that were skipped. The resulting hidden state at layer $t + \Delta t$ can be described by:

$$h_{t+\Delta t} = h_t + f(h_t). \tag{1}$$

Here the function $f(h)$ is the neural network constructed by the layers that were skipped. As outlined in [5] this formalism can be viewed as the forward Euler solution of the ODE defined by

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = f(h(t)). \tag{2}$$

Numerical solutions of the equation above can be approximated using conventional ODE solvers. When the Euler method is applied to NODEs, they behave as ResNets: the stepsize used in this method approximates discrete jumps in the network.

Common numerical solvers include the Euler method (finite difference method) and the Runge-Kutta method. Most solvers assume an initial value problem, where the initial condition (i.e. feature values) and the differential equation (i.e. neural network) are well defined. Solutions to initial value problems must be unique - their solution trajectories must not intersect. This restriction can cause differentials to become more variable, rendering them more difficult to approximate [4]. Numerical ODE solvers can be divided into fixed and adaptive stepsize algorithms. With adaptive stepsize methods, the error for given point's solution is minimized by modifying number of points used in each approximation. Linear solutions are simpler to calculate and require less steps than non-linear solutions. As solutions become more complex, the number of function evaluations (NFEs) required increases and the computational cost becomes higher. To reduce the numerical constrictions on solution trajectories, Dupont *et al.* (2019) propose to augment the dimensionality of the model's input space such that solution trajectories can follow simpler, more linear flows that require less NFEs [4].

[2] `https://github.com/BeeQC/ANODE-reproducibility`

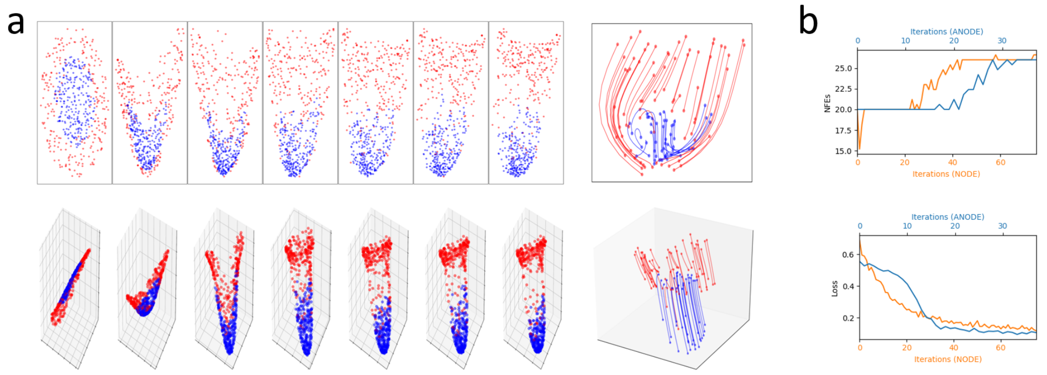

Figure 1: a. Evolution of the feature space (left) and learned flow (right) for Neural Ordinary Differential Equations (NODEs, top) and Augmented NODEs (bottom). b. Number of function evaluations (NFEs, top) per batch iteration and loss (bottom) for NODEs (bottom axis, orange) and ANODEs (top axis, blue). Note that NODEs required twice as many iterations to acheive similar performance to ANODEs.

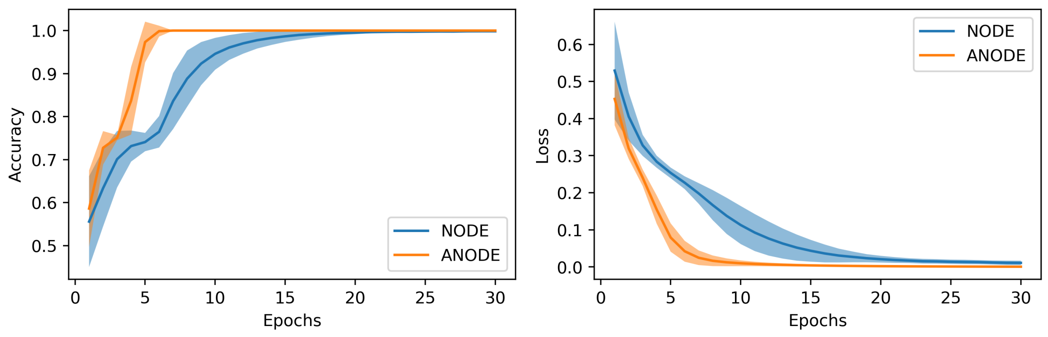

Figure 2: Accuracy (left) and loss (right) on the concentric spheres dataset. Solid lines represent the average over 100 repetitions and the shaded area represents the standard deviation.

3 Results

Dupont *et al.* (2019) compare the performance of NODE and ANODE in a 1D and a 2D toy dataset. They also conducted experiments on MNIST, CIFAR10 and Tiny ImageNet datasets.

3.1 Robustness and Replication - Toy datasets

We focused our analysis on the 2D model. This toy dataset is similar to a bullseye: a circle surrounded by a ring such that points in the inner and outer regions map to different labels. The authors show that solutions approximated with NODEs generate intersecting trajectories, whereas ANODEs with one augmented dimension generate quasi-linear solutions.

3.1.1 Bullseye Dataset: Reproducibility and Robustness

The codes provided by Dupont *et al.* 2019 were simple to implement and generated results in a reasonable amount of time. Overall we found the results obtained by Dupont *et al.* to be robust to multiple parameter changes.

Repetitions: According to the authors, each experiment was reproduced 20 times. To ensure these results were representative, we ran the toy example over 200 times. Representative results for 100 iterations can be found in figure 2 and demonstrate the results are robust.

Number of points per class: The authors conduct experiments with with 1000 points in the inner circle and 2000 points in the outer ring. We were able to reproduce results with balanced datasets (1500 points each) and various unbalanced datasets (data not shown).

Class overlap: In the paper, the authors report results on non-overlapping classes (inner region: [0, 0.5], outer region: [1, 1.5]). We conducted several experiments with overlapping regions and found that, even though ANODE seems to outperform NODE in all settings, they also struggle to perform when the overlap extends beyond 50%. An example for 33% overlap can be seen in figure 1.

Hyper-parameters: We also tested different learning rates (0.001-0.0005) and epoch sizes (1,3,10,20) and found that ANODEs outperformed NODEs in all trials (data not shown).

Others: We collected accuracy measures for all tasks; results were consistent with those observed for loss and are not included in the report. Finally, we attempted to test all three models in a new toy dataset, interleaving half-circles[3], but were unable to finalize the experiment.

3.1.2 Fixed Stepsize: RK4

ANODEs augment the space on which the neural ODE is solved. This allows numerical solvers to use these dimensions and compute simpler flows. Dupont et. al [5] demonstrate this idea by considering a toy example of two concentric spheres. While NODEs compute flows that visibly

[3]`https://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_moons.html`

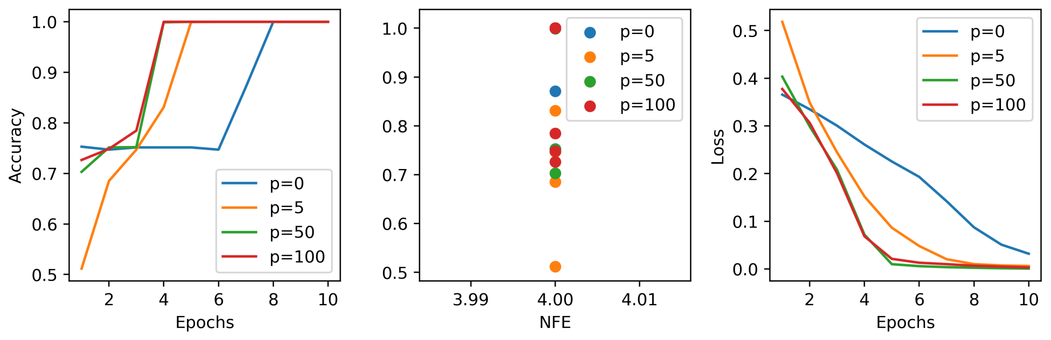

Figure 3: Performance comparison of different augmented dimensions when using a 4th order Runge-Kutta method (RK4) to solve the ANODE. (Left) Using more dimensions causes the RK4 method to achieve high accuracies with fewer epochs. (Center) RK4 uses a fixed number of function evaluations. (Right) ANODEs solved with RK4 achieve lower losses with more augmented dimensions.

contain high curvature, an ANODE is capable of separating the point using quasi-linear flows. Due to complex curvature in the solution trajectories that NODEs need to estimate, adaptive stepsize methods are required. However, if these flows were quasi-linear, fixed-step methods should also be capable of achieving high accuracies. We tested this hypothesis by using the 4th order Runge-Kutta method (RK4), which is a fixed-step method and uses a constant number of function evaluations when computing solution trajectories. Using this method, we fit the model to the toy data-set, while adjusting the number of augmented dimensions. We find that fewer epochs are required to achieve high accuracies and low losses when more augmented dimensions are used (Figure 3). This finding supports the claim that dimension-augmentation decrease solution complexity.

3.2 Robustness and Replication - Image datasets

Due to computational limitations, our image experiments were performed on the MNIST dataset alone. As described by Dupont et. al [5], ANODEs perform better than NODEs when being trained, achieving higher accuracy and lower loss in a given number of epochs 4. Furthermore ANODEs require less function evaluations, implying that they are also faster at computing flows.

3.2.1 MNIST dataset: number of dimensions

In theory, increasing the number of augmented dimensions should make ANODEs easier to solve and require less NFEs. However, as discussed by Dupont et al [5], increasing the number of augmented

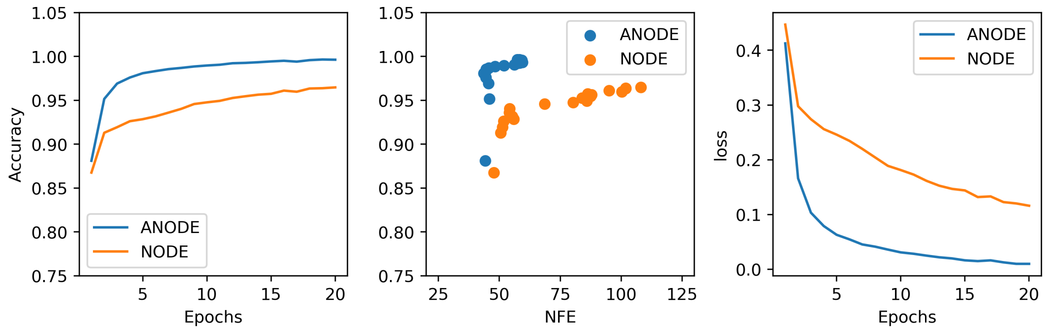

Figure 4: Comparison of ANODEs and NODEs when training on the MNIST dataset. This ANODE uses 5 augmented dimensions and three 2D convolutional layers, as described in [5]. (Left) ANODEs achieve high training accuracies faster than NODEs. (Center) The ANODE uses less function evaluations than the NODE to achieve high accuracies. (Right) ANODE achieves lower losses than NODE in a given number of epochs.

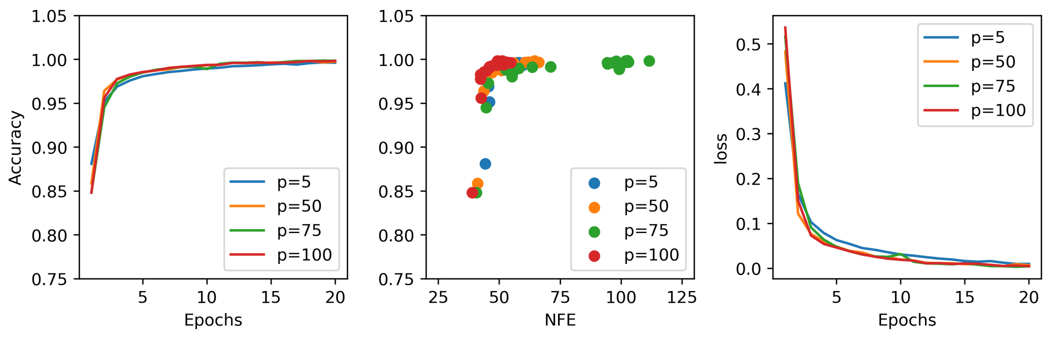

Figure 5: Evaluating the effect of increasing augmented dimensions on training performance. (Left) ANODES achieve comparable accuracies when using different augmented dimensions, with a slight increase in performance when using 100 augmented dimensions. (Center) The ANODE uses less function evaluations than the NODE to achieve high accuracies. (Right) ANODE achieves lower losses than NODE in a given number of epochs.

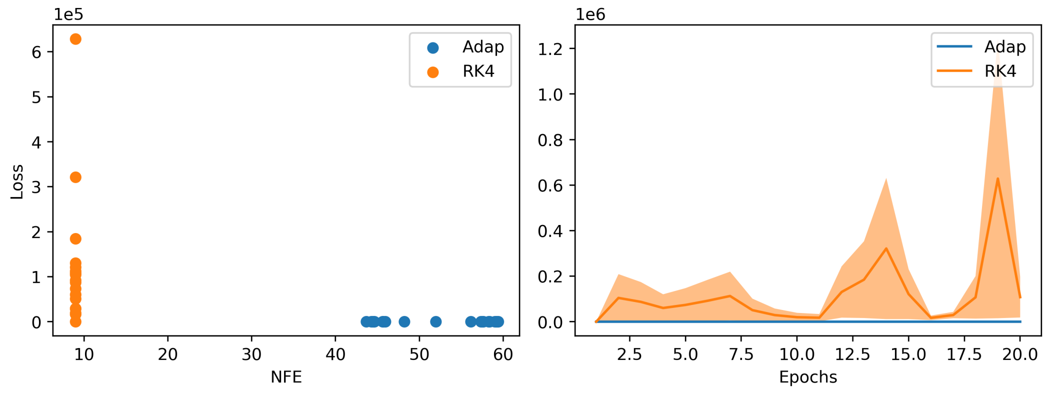

Figure 6: Comparison between stepsize adaptive methods and RK4 when training on MNIST dataset. (Left) RK4 uses constant NFEs since it is a fixed step size method. (Right) RK4 is not able to achieve low losses in a stable manner.

dimensions can cause ANODEs to become unstable. To test these hypotheses, we treated augmented dimensions as hyper-parameter when training ANODEs on the MNIST dataset, and evaluated training performance with different numbers of augmented dimensions. Generally we find that training accuracy and loss are comparable when using more augmented dimensions (Figure 5, left and right panels). At 100 augmented dimensions ANODEs used less NFEs than when using fewer augmented dimensions. However, using 75 augmented dimensions required many more NFEs compared to the NFEs required when training with 5 dimensions (Figure 5, center). It should be noted that due to our restrictions in computational resources, only one replicate of this data was collected.

When computing flows for large data sets like MNIST, solution trajectories are expected to be complex and require several function evaluations to be approximated. Adaptive stepsize methods are suitable for these scenarios as stepsize would be reduced (leading to increments in NFEs) when solution trajectories become more complicated. Therefore we hypothesise that ablating the ability to adapt stepsize would lead to unstable loss. To test this hypothesis, we replace the adaptive step size method used in [5] by a 4^{th} order Runge-Kutta method (RK4), which uses a fixed step size, i.e. fixed NFEs, when computing the next point in a solution trajectory. Figure 6 (left) displays that RK4 uses a fixed number of function evaluations while adaptive methods do not. While this would lead to high computational efficiency, using RK4 leads to unstable and high-magnitude losses when compared to adaptive stepsize methods for highly complex solution trajectories (Figure 6, right).

3.3 Effect of Filter Size on ANODE/NODE Performance on Image Tasks

The original authors carried out their image experiments with filter sizes of 64 and 92 for ANODE and NODE respectively, with the aim of maintaining an approximate number of parameters across both models (Dupont et al., F.2.2) [5]. As one of the few hyper-parameters for which no principled search is described, we explore here a small range of filter sizes (limited by computational time).

ANODE and NODE models displayed similar convergence with filter sizes in the range of 32 to 92, with slightly more variation in the ANODE (Figure 7, Top Row). The total number of functions evaluated trended similarly for the NODE models across all filter sizes trained (Figure 7, Bottom Row). Filter sizes of 64 and 92 accumulated a similar number of operations over time, a filter size of 32 consistently outperforms the other models tested, despite converging in a similar fashion. This is illustrated when comparing the total number of functions evaluated directory to the training loss (Figure 8). Furthermore, there is far less variability observed in the number of functions evaluated amongst NODE models tested, while much variability is observed in the ANODE models.

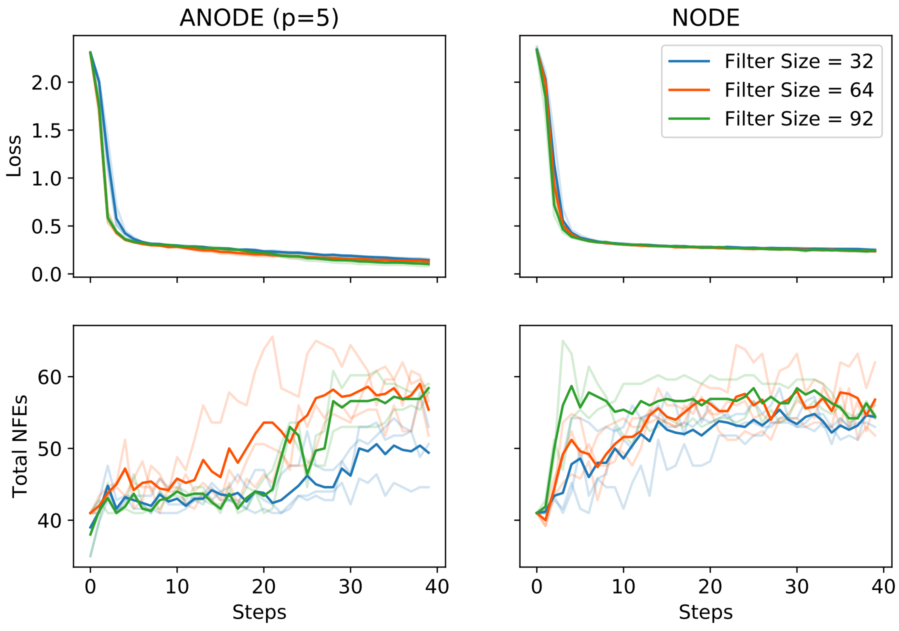

Figure 7: Loss (y-axis) and total number of functions evaluated (x-axis) of ANODE ($p = 5$, Left Column) and NODE (Right Column) experiments, trained on the MNIST task, with filter sizes ranging from 32 to 92. Faint traces are replicates while bold ones are averages of the replicates.

4 Discussion

We were able to reproduce most major results presented in the original study using the authors implementation of ANODEs. We established model robustness by repeating experiments multiple times and changing hyperparameters (Figure 2). We used ANODE to train a binary classifier with concentric spheres with various degrees of overlap. ANODEs outperformed NODEs in this task while using less function evaluations, demonstrating how flow trajectories are simplified compared to the non-augmented NODE (Figure 1, panel **b**). When NODEs were trained with double the number of epochs, however, both algorithms performed equally well showing the robustness of the algorithm (Figure 1, panel **b**). To further demonstrate how augmentation simplifies solution trajectories, we used fixed stepsize ODE solvers while increasing the number of augmentation dimensions (Figure 3). We found that performance increased with more augmented dimensions, implying that solutions have become easier to compute.

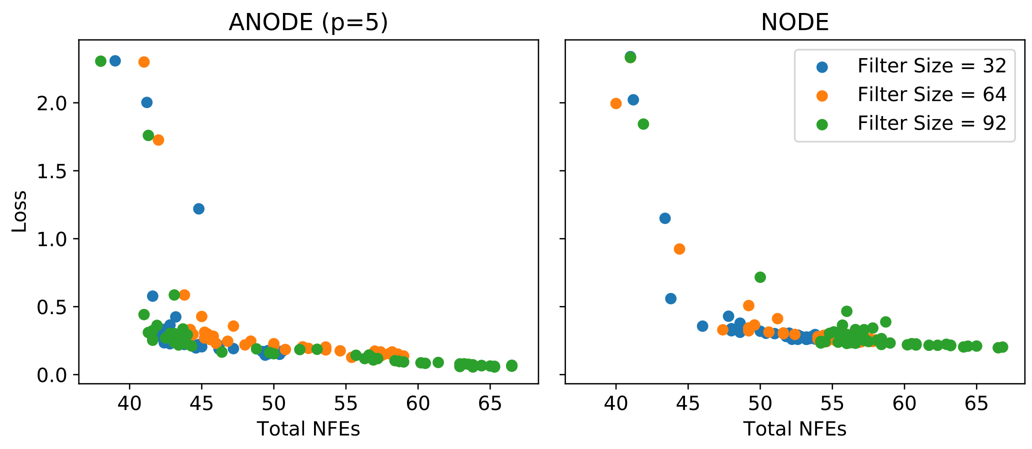

Figure 8: Loss (y-axis) and total number of functions evaluated (x-axis) of ANODE ($p = 5$, Left Column) and NODE (Right Column) experiments, trained on the MNIST task, with filter sizes ranging from 32 to 92. Points represent averages over three replicates.

Generally, we found that ANODEs require less function evaluations when trained on the MNIST data set (Figure 4), achieving lower loss compared to NODEs given a fixed number of epochs. One primary concern discussed in [5] is that ANODEs performed worse when excessively many dimensions were used to augment the NODE. We used augmentations ranging from 5 to 100 dimensions when training on the MNIST dataset. Generally we found that ANODEs still perform with high training accuracies and stable loss dynamics. However the NFEs for such augmentations seemed to fluctuate, with over 100 NFEs when using 75 dimensions for augmentation (Figure 5). It should be noted that these results stem from only one replicate of results. Given more time and computational resources, more replicates should be collected.

Hyperparameters were chosen by the original authors in a principled manner, but the number of filters chosen for the MNIST image model was not commented on when considering the number of model parameters. We explored a small range of filter sizes and observed small change in performance. However, some of our results indicate that this could be an interesting topic for further investigation.

5 Conclusion

Augmented Neural ODEs by Dupont et. al [5] identifies and targets a central limitation of Neural Ordinary Differential Equations: solution trajectories cannot intersect. This restriction may lead solution trajectories to become very complex, and computationally expensive to estimate. Dupont et. al further note that certain classes of problems cannot be solved by NODEs due to this restriction. By augmenting the feature space in which NODEs are solved, Dupont et. al [5] describe a new class of NODEs that is more efficient and generalizable. Using the implementation provided by Dupont et. al[5], we reproduced some of the major results presented in the article and showed that the method is robust to changes in hyperparameters and initial conditions. We also provide additional evidence that augmentation simplifies flow by considering fixed-step solving methods. Since these methods do not adapt to data complexity their performance will decrease with increased complexity. Our results demonstrate that fixed-step size algorithms perform better with increasing augmented dimensions, supporting Dupont et. al's claim that ANODEs reduce flow complexity.

Acknowledgements

We thank Dupont et. al for providing commented code and reproducible examples of their ANODE implementation. This work was supported by *NSERC Create - Complex Dynamics in Brain and Behaviour*, *NSERC Canada Graduate Scholarship - Master's* and *FRQNT - Bourse de maîtrise en recherche*.

References

[1] Monya Baker. “1,500 scientists lift the lid on reproducibility”. In: Nature News 533.7604 (2016), p. 452.

[2] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: 2016, pp. 770–778. URL: http://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html.

[3] Matthew Hutson. “Artificial intelligence faces reproducibility crisis”. In: (2018).

[4] Steven H Strogatz. “Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering”. In: (2018).

[5] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. “Augmented Neural ODEs”. In: arXiv:1904.01681 [cs, stat] (Oct. 2019). arXiv: 1904.01681. URL: http://arxiv.org/abs/1904.01681.