CS-598: Deep Generative and Dynamical Models **Spring 2023**

Topic 25: Neural Ordinary Differential Equations 2

Lecturer: Arindam Banerjee *Scribe: Nicolas Nytko, Sean Farhat, Nathanael Assefa*

25.1 Augmented Neural ODEs

25.1.1 Background

In the prior lecture, we covered Neural Ordinary Differential Equations (NODEs), a more continuous analog of the fixed step Euler Integral view of ResNets presented in the Awarded NeurIPS '18 best paper by David Duvenaud's UToronto Team[3].

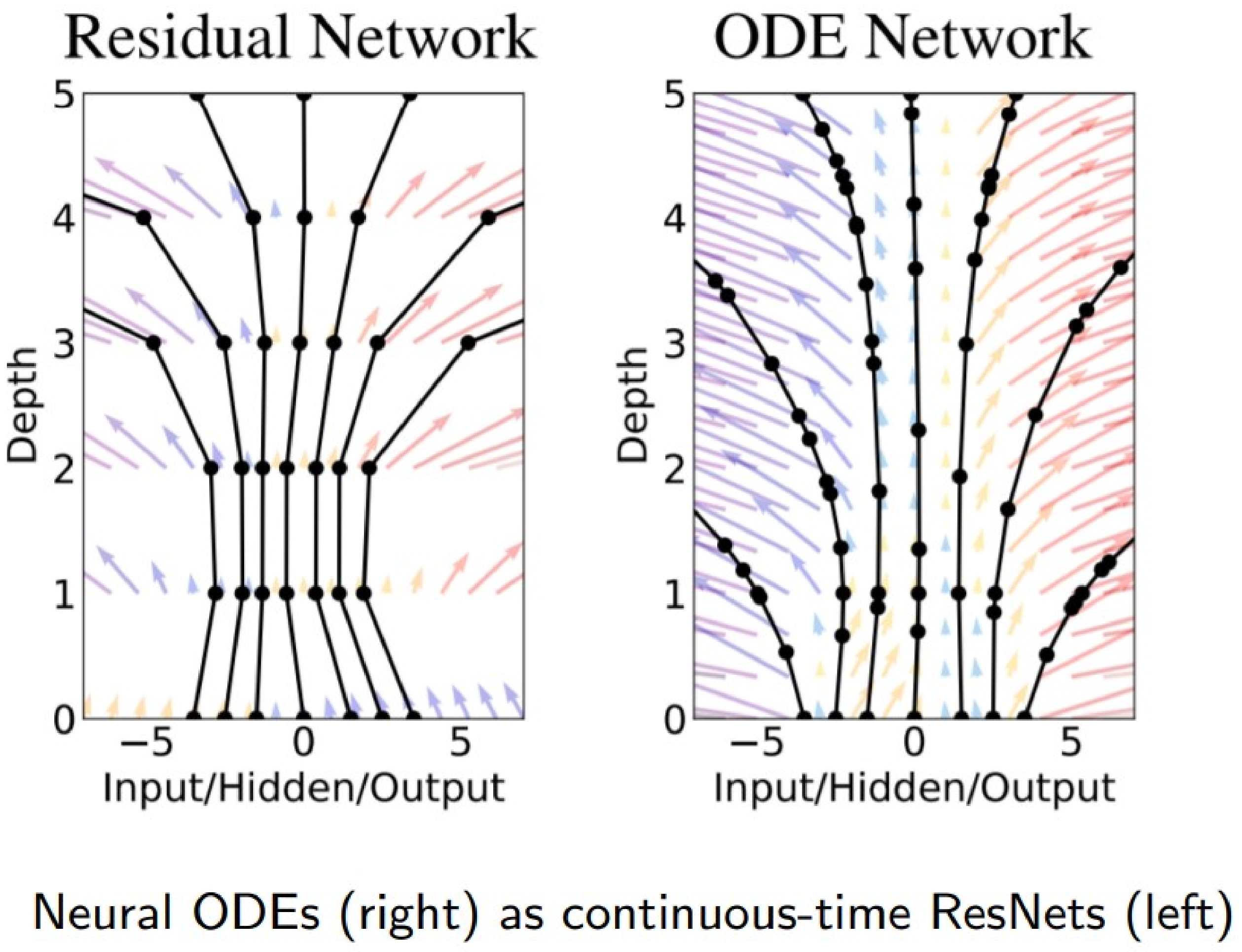


FIGURE 25.1: Lecture illustration from the Toronto NODEs Paper

We can illustrate this Euler integral perspective of recurrence in the following expression. Consider the propagation of the hidden state from a layer t to $t+1$ in ResNets as $\mathbf{h}_{t+1} = \mathbf{h}_t + \mathbf{f}_t\left(\mathbf{h}_t\right)$ where $\mathbf{h}_t \in \mathbb{R}^d$ is the hidden state at layer t. Notice that the expression can be rearranged into the equivalent expression $\lim_{\Delta t \to 0} \frac{\mathbf{h}_{t+\Delta t} - \mathbf{h}_t}{\Delta t} = \frac{\mathrm{d}\mathbf{h}(t)}{\mathrm{d}t} = \mathbf{f}(\mathbf{h}(t), t)$ via simple arithmetic. This second expression is a first-order ordinary differential equation (ODE) in explicit form. The corresponding initial condition (IC) of this ODE is $\mathbf{h}(0) = \mathbf{x}$ as a matter of technicality because the input layer takes in the input vector $\mathbf{x}$.

To make the ResNets analogy more explicit, we can map an input $\mathbf{x}$ to some output $\mathbf{y}$ by a forward pass of the neural network adjust the weights of the network to match $\mathbf{y}$ with some $\mathbf{y}_{\text{true}}$. In NODEs, we map an input $\mathbf{x}$ to an output $\mathbf{y}$ by solving an ODE starting from x. We then adjust the dynamics of the system (encoded by f) such that the ODE transforms x to a $\mathbf{y}$ which is close to y_{true} .

25.1.2 NODEs for Continuous and Discontinuous Functions

The author's term the vector field approximated by the NODE the flow $\phi_t : \mathbb{R}^d \to \mathbb{R}^d$ is defined as the hidden state at time t, i.e. $\phi_t(\mathbf{x}) = \mathbf{h}(t)$, when solving the ODE from the initial condition $\mathbf{h}(0) = \mathbf{x}$.

Although we can use ODEs to model flows $\phi(\mathbf{x}) \in \mathbb{R}^d$ we may be interested in learning classification functions from $\mathbb{R}^d$ to $\mathbb{R}$. The authors define a NODE analog to Lin & Jegelka's (2018) ResNets model for this task, producing a simple model architecture: an ODE layer, followed by a linear layer; defining a model from $g(\mathbf{x}) = \mathcal{L}(\phi(\mathbf{x}))$ where $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$.

25.1.3 NODEs Approximation Counter Examples

The authors present functions that NODEs cannot represent, motivating their work.

Counter Example 1: Let $g_{1\ \mathrm{d}} : \mathbb{R} \to \mathbb{R}$ be a function such that $g_{1\ \mathrm{d}}(-1) = 1$ and $g_{1\ \mathrm{d}}(1) = -1$. The flow of an ODE cannot represent $g_{1d}(x)$.

The authors note the puzzling situation that if NODEs can be interpreted as continuous equivalents of ResNets, so it is interesting to consider why ResNets can represent $g_{1\ \mathrm{d}}(x)$ but NODEs cannot. They note that this is due to an artifact of the Poincaré–Miranda theorem, or specifically that ab intermediate value of the flow vector field associated with $\mathbf{f}(\mathbf{h}(t), t)$ of the ODE cannot be unique and bijective in this expression because it must intersect at an intermediate value.

Counter Example 2: Let $0 < r_1 < r_2 < r_3$ and let $g : \mathbb{R}^d \to \mathbb{R}$ be a function such that

$$\begin{cases} g(\mathbf{x}) = -1 & \text{if } \|\mathbf{x}\| \le r_1 \\ g(\mathbf{x}) = 1 & \text{if } r_2 \le \|\mathbf{x}\| \le r_3 \end{cases}$$

increasing number of function evaluations (NFE) where $\|\cdot\|$ is the Euclidean norm. The function maps all points inside the blue sphere to -1 and all points in the red annulus to 1 . Neural ODEs cannot represent $g(\mathbf{x})$.

25.1.4 NODEs Approximation Limits

NODEs approximate homeomorphisms : The feature mapping $\phi(\mathbf{x})$ is a homeomorphism, so the features of Neural ODEs preserve the topology of the input space.

The authors prove this in the appendix by noting that NODEs are continuous and invertible and as a consequence of the flow of an ODE is a homeomorphism, i.e. a continuous bijection whose inverse is also

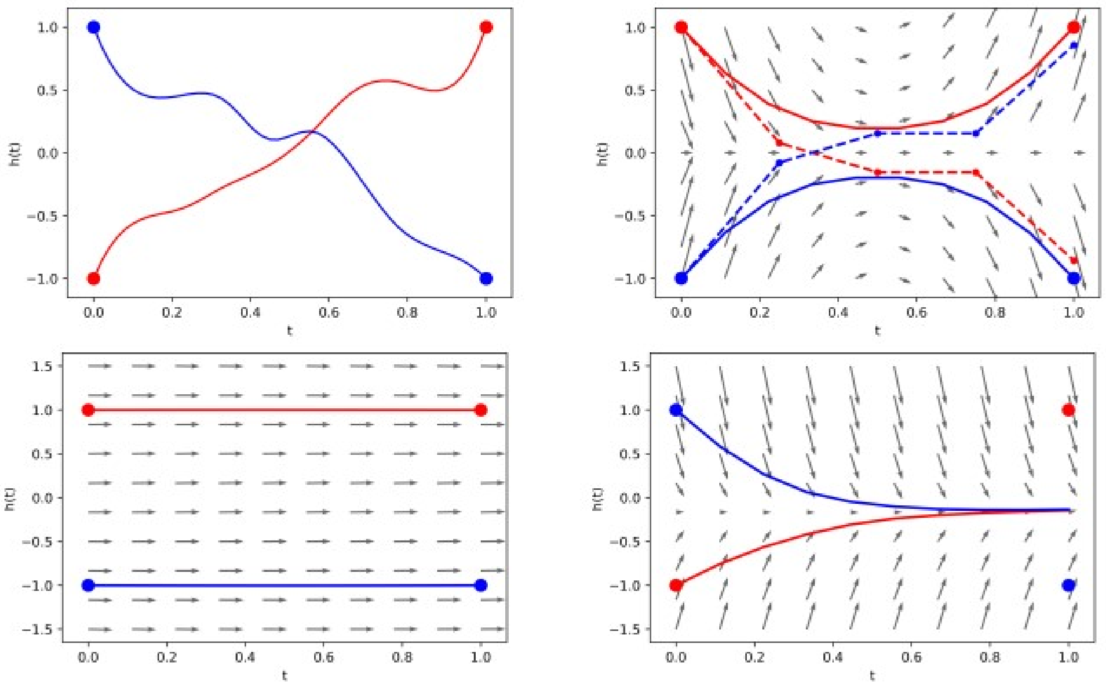
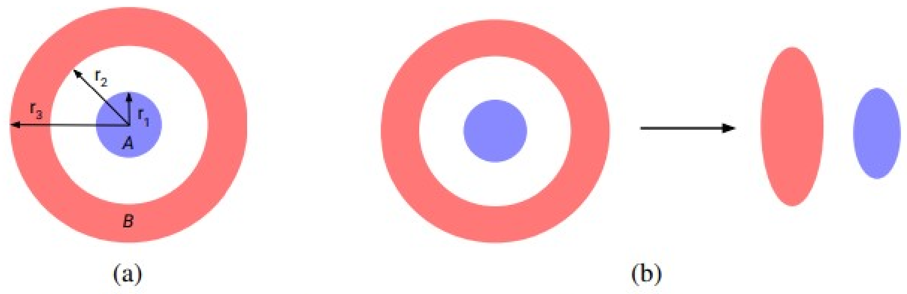


FIGURE 25.2: **Illustration of Counter Example 1**. Top Left: Possible ideal path unrestricted function mapping graph for g_{1d}. Top Right: The gradient vector field and possible paths of the function's mapping in NODEs, demonstrate the inevitable crossing and zero gradient. Bottom Left: Satisfactory uniform field for linear continuous identity function (is homomorphic). Bottom Right: Failure of NODE gradient vector field to reconcile crossing and realize mapping.



(a) (b)

Figure 15: (a) Diagram of $g(\mathbf{x})$ in 2d. (b) An example of the map $\phi(\mathbf{x})$ from input data to features necessary to represent $g(\mathbf{x})$ (which NODEs cannot learn).

FIGURE 25.3: **Illustration of Counter Example 2**. The function $g(x)$ maps all the points in the blue circle to -1 and the red ring to 1 . Left: 2D Top Down. Right: 2D Cross Section.

continuous. This implies that NODEs can only continuously deform the input space and cannot separate a connected region of space.[1]

Noting this proposition the authors run a series of level experiments with the ResNet and NODE (Figure 25.3) to evaluate if smoothing the function space by increasing the size of the NODE network can better approximate the discontinuous function as a homeomorphism. The experiments produce an increasing number of function evaluations (NFE) with little improvements that the authors note is akin to an ill-posed ODE problem. [4]

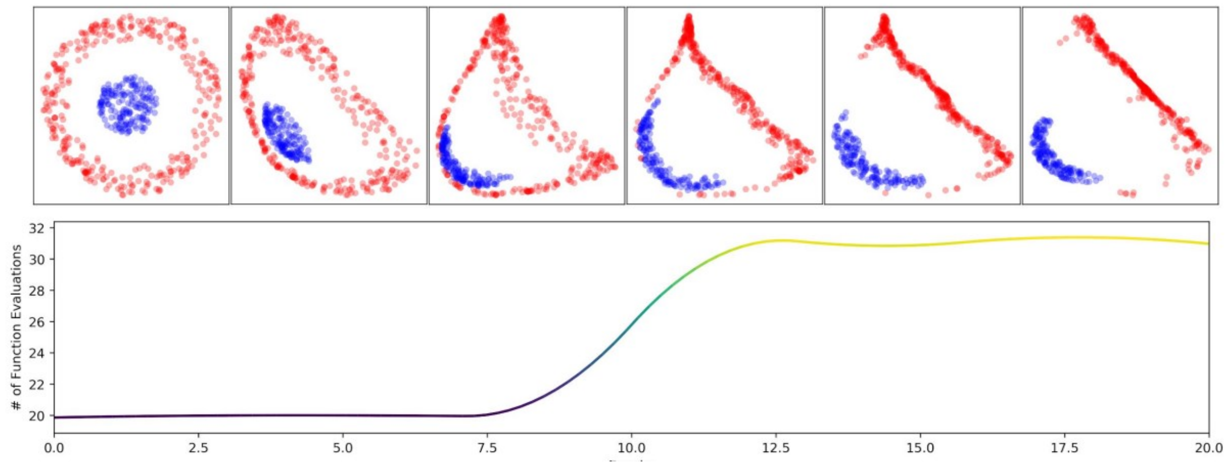


FIGURE 25.4: Increasing number of function evaluations (NFE) in approximating a 3D non-homeomorphic function

25.1.5 Augmented NODEs (ANODE)

The authors then propose the following approach that they term the Augmented NODE (ANODE):

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\mathbf{h}(t)\\ \mathbf{a}(t)\end{bmatrix} = \mathbf{f}\left(\begin{bmatrix}\mathbf{h}(t)\\ \mathbf{a}(t)\end{bmatrix}, t\right), \quad \begin{bmatrix}\mathbf{h}(0)\\ \mathbf{a}(0)\end{bmatrix} = \begin{bmatrix}\mathbf{x}\\ \mathbf{0}\end{bmatrix}$$

Lifting the ODE from $\mathbb{R}^d$ to $\mathbb{R}^{d+p}$ additional dimensions to avoid trajectories intersecting each other. Letting $\mathbf{a}(t) \in \mathbb{R}^p$ denote a point in the augmented part of the space, we can formulate the augmented ODE problem as we concatenate every data point x with a vector of zeros and solve the ODE on this augmented space. An additional smoother, f, is applied to give simpler flows that the ODE solver can compute in fewer steps.

25.1.6 ANODE Discussion

The authors run similar experiments to the NODE on classification type problems and demonstrate ANODEs can model more complex functions using simpler flows while achieving lower losses, reducing computational cost, and improving stability and generalization where NODEs lead to slower learning and complex flows which are expensive to compute.

The authors note that the success of the lifting procedure for augmenting NODEs leaves a few areas for thought.

- ANODEs may not always be appropriate as increasing the dimension of a function may be undesirable for specific applications.
- The augmented dimension can be seen as an extra hyper-parameter to tune and the selection process isn't optimized.
- While the concatenation of at least 2 vectors in the lift yields benefits in binary the classification considered since NODEs only learn features that are homeomorphic to the input space, excessively large augmented dimensions (e.g. adding 100 channels to MNIST), the model tends to perform worse with higher losses and NFEs.
- They authors call for further analysis on why lifting augments NODEs for bounding the range in the former item and propose also exploring noising techniques for adapting NODEs for classification problems.

25.1.7 ANODE Results

The authors have extensive well visualized experimental results illustrating the effectiveness of augmenting the NODE. The figure (25.5) below highlights which value each point in the input space gets mapped by an optimized NODE and an optimized ANDOE. Because NODEs deform spaces continuously the learned flow is smeared through the annulus while the ANODEs map the points accurently and with better generalization.

The next set of figures, (25.6) and (25.7), demonstrate the accuracy improvements and reduced computation derived from the ANODEs design to learn simpler flows. All these results are highlighted at the start of the prior section.

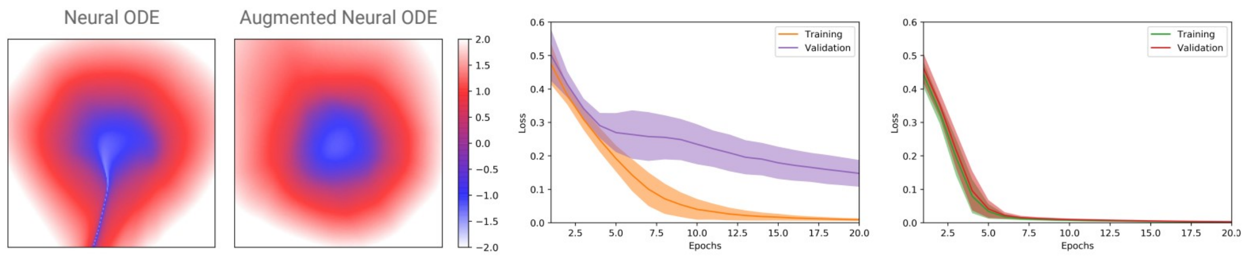


Figure 25.5: Plots of how ANODEs Generalize: (Left) Plots of how NODEs and ANODEs map points in the input space to different outputs (for zero training loss). ANODEs generalize better. (Middle) Training and validation losses for NODE, (Right) for ANODE

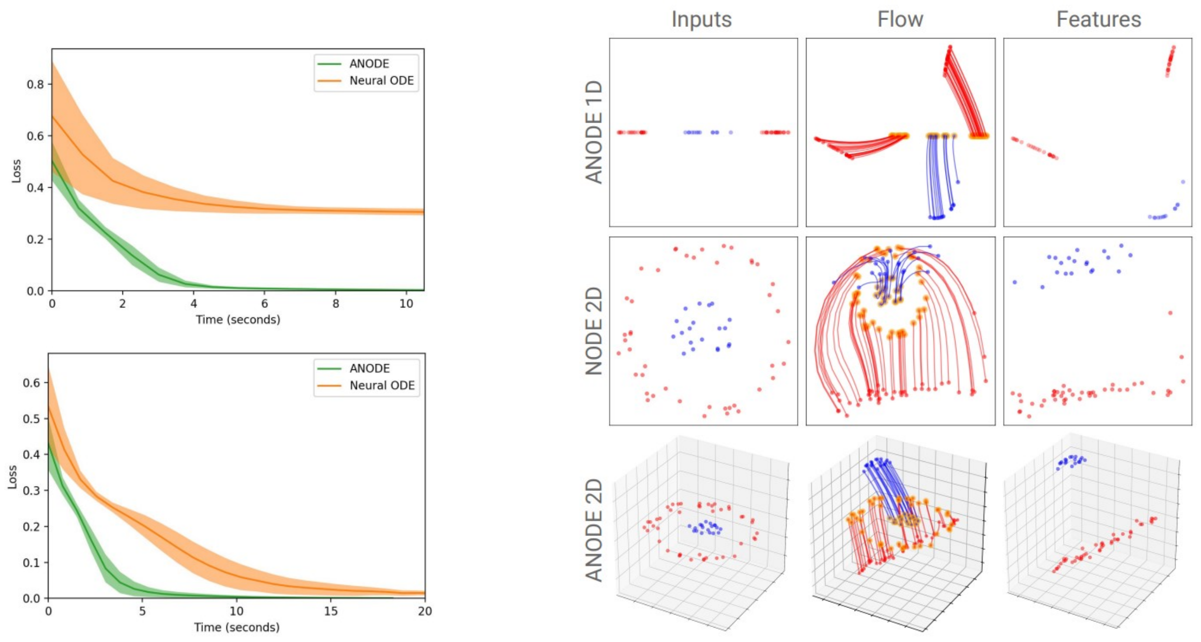


Figure 25.6: Loss function for NODEs and ANODEs trained on d=1 (top) and d=2 (bottom). ANODEs are faster. On the right are the flows learned by NODEs and ANODEs. ANODEs learn simple nearly linear flows, while NODEs learn complex (expensive) flows

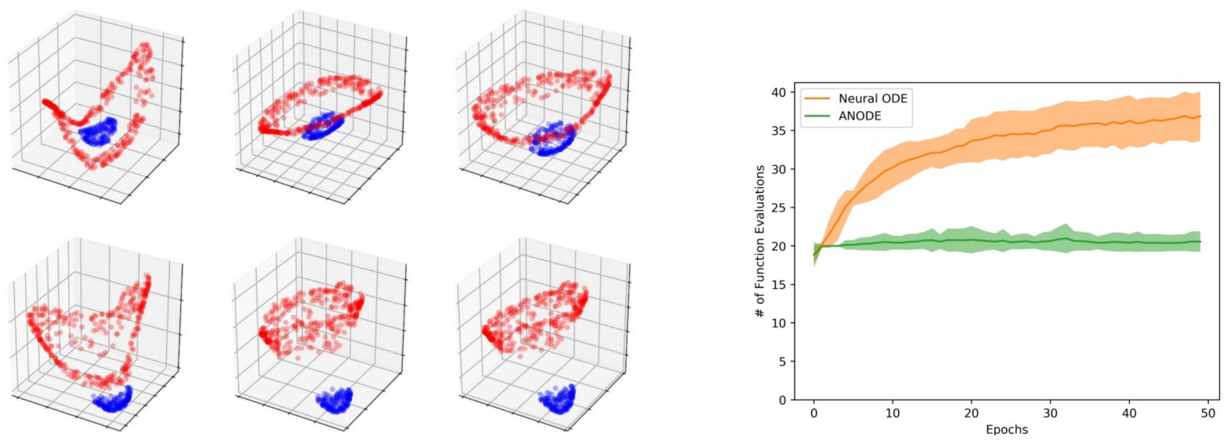


FIGURE 25.7: Evolution of features during training for ANODEs. The top left pane shows the feature space for a randomly initialized ANODE and the bottom right shows the features after training. The right panel shows the evolution of NFEs during training

25.2 Approximation Capabilities of Neural ODEs and Invertible Residual Networks

25.2.1 Connection to ANODEs

This is a theoretical work directly addressing the last bullet in the page above regarding the dimension of the sufficent dimension lift, as acknowledged in the paper's *"Our Contribution"* section. The authors show that NODEs and an i-ResNets–a ResNet with a bounded Lipschitz of the activation less than 1 to ensure invertibility, followed by a single linear layer, can approximate functions, including non-invertible functions, equally well as any traditional feedforward neural network. Since feed-forward networks that are shallow-but-wide or narrow-but-deep unconstrained ResNet-based architecture are universal approximators, so are ODE-Nets and i-ResNets.[2]

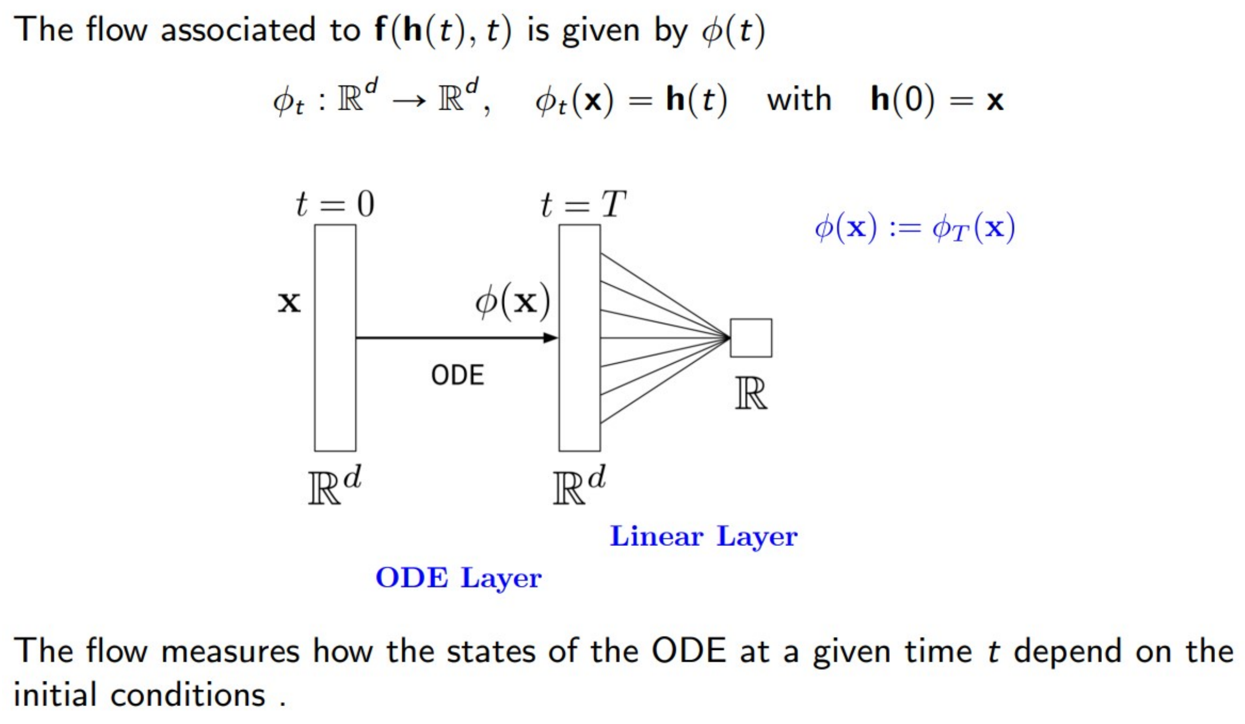
The authors present both Neural ODEs and i-ResNet as recently proposed methods for enforcing invertibility of residual neural models. The authors then apply a set of modifications to produce a generic technique for modeling homeomorphisms by first noting the limited approximation capabilities of Neural ODEs and i-ResNets. Finally, they conclude by showing that a Neural ODE or an i-ResNet with a single linear layer is sufficient to turn the model into a universal approximator for non-invertible continuous functions.[5]

25.2.2 Approach

The authors begin the analysis by citing the results of the ANODE authors that focus on a p-ODE-Net followed by a linear layer and reference the counterexamples and the subsequent discussions above. They lay the claim that setting $q = p + 1$ is enough to turn a NODE followed by a linear layer into a universal approximator $\mathbb{R}^p \to \mathbb{R}$ by modeling invertible functions - homeomorphisms - via analyzing pure ODENets and i-ResNets. They begin by laying a similar but more generic counter-example in their first theorem.

Theorem 1: Let $\mathcal{X} = \mathbb{R}^p$, and let $\mathcal{Z} \subset \mathcal{X}$ be a set that partitions $\mathcal{X}$ into two or more disjoint, connected subsets C_i, for $i = [m]$. Consider a mapping $h : \mathcal{X} \to \mathcal{X}$ that

- is an identity transformation on $\mathcal{Z}$, that is, $\forall z \in \mathcal{Z}, h(z) = z$
- maps some $x \in C_i$ into $h(x) \in C_j$, for $i \neq j$.


The flow associated to $\mathbf{f}(\mathbf{h}(t), t)$ is given by $\phi(t)$

$$\phi_t : \mathbb{R}^d \to \mathbb{R}^d, \quad \phi_t(\mathbf{x}) = \mathbf{h}(t) \quad \text{with} \quad \mathbf{h}(0) = \mathbf{x}$$

The flow measures how the states of the ODE at a given time t depend on the initial conditions .

FIGURE 25.8: Flow Definition Graphic from Lecture

Then, no p-ODE-Net can model h, showing of $\mathcal{X} \to \mathcal{X}$ invertible mappings that cannot be expressed by these modeling approaches when they operate within $\mathcal{X}$. They then prove that any homeomorphism $\mathcal{X} \to \mathcal{X}$, for $\mathcal{X} \subset \mathbb{R}^p$, can be modeled by a Neural ODE / i-ResNet operating on an Euclidean space of dimensionality $2p$ that embeds $\mathcal{X}$ as a linear subspace.

25.2.3 ANODE Approximation Proof

The authors construct a mapping from the original problem space, $\mathcal{X} \in \mathbb{R}^p$ into $\mathbb{R}^{2p}$ that suffices two conditions, (1) preserves $\mathcal{X}$ as a p-dimensional linear subspace consisting of vectors and (2) leads to an ODE that maps $\left[x, 0^{(p)}\right] \to \left[h(x), 0^{(p)}\right]$.

This provides a solution with a structure that is convenient for out-of-the-box training and inference using NODEs. If the NODEs operates on Euclidean space of dimensionality $q > p$, we can approximate arbitrary p-homeomorphism $\mathcal{X} \to \mathcal{X}$, as long as q is high enough. Under the constructed mapping above, it suffices to take $q = 2p$ via adding p zeros to input vectors. This produces the ANODE range bound mentioned in the last item of the future work sections in the author's Theorem 2.

Theorem 2: For any homeomorphism $h : \mathcal{X} \to \mathcal{X}, \mathcal{X} \subset \mathbb{R}^p$, there exists a $2p - ODE - Net$ $\phi_T : \mathbb{R}^{2p} \to \mathbb{R}^{2p}$ for $T = 1$ such that $\phi_T\left(\left[x, 0^{(p)}\right]\right) = \left[h(x), 0^{(p)}\right]$ for any $x \in \mathcal{X}$.

This theorem is proven constructively by showing a vector field in $\mathbb{R}^{2p}$ with the desired properties. The authors analyze the extent to which the extra p dimensions can ensure the bijectivity of the paths in this construction, a sample delta function is used and operations are enumerated along the network.

Intuitively, the result is an artifact of the ODE variability with respect to time and the discrete representation in vector form. It is possible to always find a nonintersecting path representation via the NODE of 2p for the

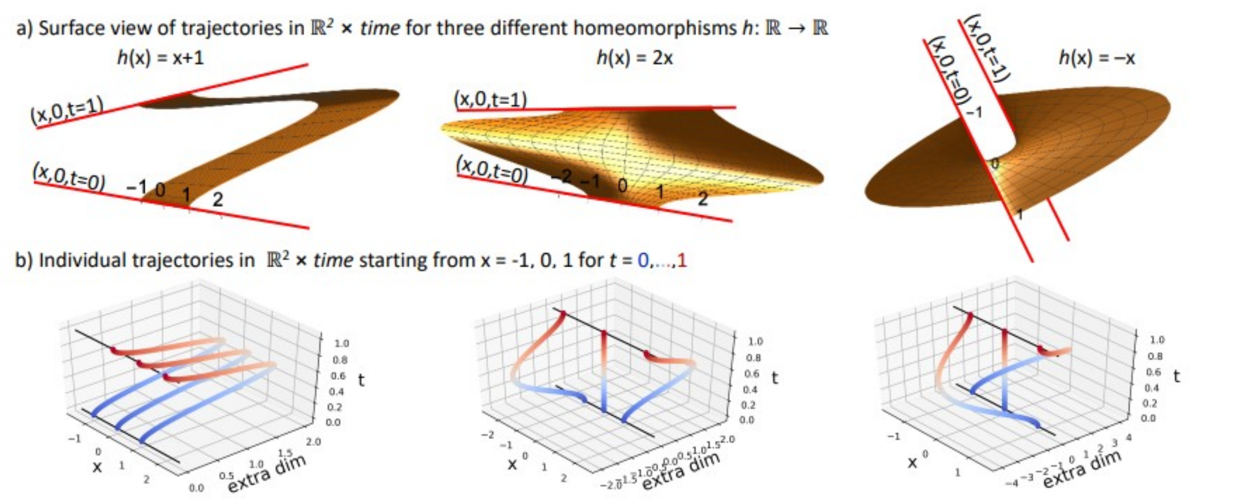


FIGURE 25.9: Trajectories in $\mathbb{R}^{2p}$ that embed an $\mathbb{R}^p \to \mathbb{R}^p$ homeomorphism, using $f(\tau) = (1 - \cos \pi\tau)/2$ and $g(\tau) = (1 - \cos 2\pi\tau)$. Three examples for $p = 1$ are shown, including the mapping $h(x) = -x$ that cannot be modeled by Neural ODE on $\mathbb{R}^p$, but can in $\mathbb{R}^{2p}$.

p-dimensional homeomorphism due to the single differentiable variable of the ODE. The authors present the paper's sole figure (25.9) to illustrate this trajectory relaxation.

Based on the above result, we now have a simple method for training a Neural ODE to approximate a given continuous, invertible mapping h and its continuous inverse h^{-1}, provided each sample x is augmented with p zeros.

25.2.4 Core Result

The authors provide this last general result in their final Theorem 7 that NODEs and iResNets are universal approximators, and the expression is implemented by adding a linear layer for lifting/projecting the dimensions of the ANODEs/2p-ODE-Net into the input/output dimensions p/r.

Theorem 7: Consider a neural network $F : \mathbb{R}^p \to \mathbb{R}^r$ that approximates function $f : \mathcal{X} \to \mathbb{R}^r$ that is Lebesgue integrable for each of the r output dimensions, with $\mathcal{X} \subset \mathbb{R}^p$ being a compact subset. For $q = p + r$, there exists a linear layer-capped q-ODE-Net that can perform the mapping F. If f is Lipschitz, there also is a linear layer-capped q-i-ResNet for F.

25.2.5 Experimental Results

The paper runs experiments on CIFAR10 using an off-the-shelf ResNet implementation and the ANODE from the first paper, and it validates the theorems according to their results, shown below.

References

[1] Stephen A Andrea. On homeomorphisms of the plane, and their embedding in flows, 1965.

[2] Jens Behrmann, Will Grathwohl, Ricky T. Q. Chen, David Duvenaud, and Jörn-Henrik Jacobsen. Invertible residual networks, 2019.

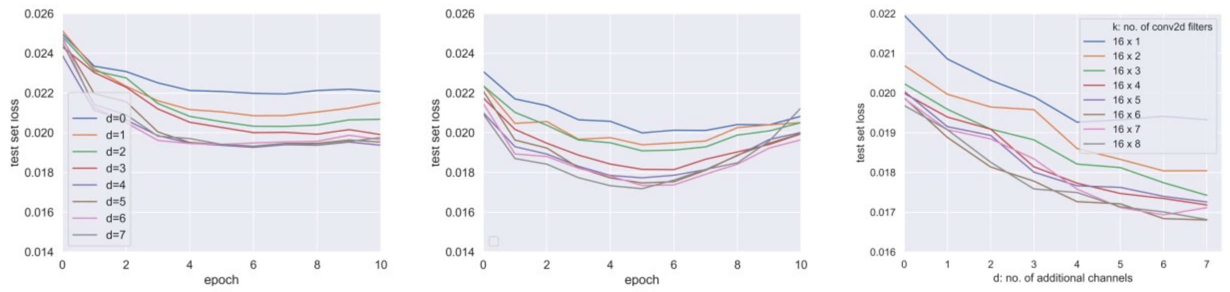


FIGURE 25.10: Left and center: test set cross-entropy loss, for increasing number d of null channels added to RGB images. For each d, the input images have dimensionality $32 \times 32 \times (3 + d)$. Left: ODE-Net with k=64 convolutional filters; center: k=128 filters. Right: Minimum of test set cross-entropy loss across all epochs as a function of d, the number of null channels added to input images, for ODE-Nets with different number of convolutional filters, k.

[3] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations, 2019.

[4] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes, 2019.

[5] Han Zhang, Xi Gao, Jacob Unterman, and Tom Arodz. Approximation capabilities of neural odes and invertible residual networks, 2020.