Invertible Residual Networks

Jens Behrmann[1,2], David Duvenaud[1], Jörn-Henrik Jacobsen [1]
[1]Vector Institute and University of Toronto
[2]University of Bremen, Center for Industrial Mathematics
jensb@uni-bremen.de, {duvenaud,jjacobs}@cs.toronto.edu

Abstract

Reversible deep networks provide useful theoretical guarantees and have proven to be a powerful class of functions in many applications. Usually, they rely on analytical inverses using dimension splitting, fundamentally constraining their structure compared to common architectures. Based on recent links between ordinary differential equations and deep networks, we provide a sufficient condition when standard ResNets are invertible. This condition allows unconstrained architectures for residual blocks, while only requiring an adaption to their regularization scheme. We numerically compute their inverse, which has $\mathcal{O}(1)$ memory cost and computational cost of 5-20 forward passes. Finally, we show that invertible ResNets perform on par with standard ResNets on classifying MNIST and CIFAR10 images.

1 Introduction

Reversible deep networks are one of the first classes of deep networks that provide statistical guarantees and access to key quantities like densities and gradients for generative modelling. Most notably, they allow for the exact computation of the determinant of their Jacobian, crucial for maximum-likelihood based generative models [10, 17, 7]. Further, they provide guarantees on mutual information between input and intermediate outputs, making them a suitable tool to analyze invariants of learned representations [16, 1]. The ability to invert deep networks alleviates the need to store intermediate activations during backpropagation [11, 6] and offers a promising approach in Inverse Problems [3]. However, architectures used to achieve reversibility are typically restricted to have analytical inverses [9].
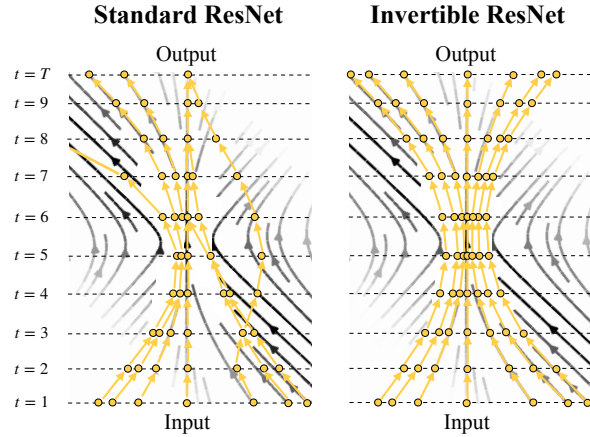

Figure 1: *Left:* Standard ResNets do not preserve invertibility of underlying continuous dynamic as paths cross and collapse. *Right:* Invertible ResNets approximate underlying dynamic, paths do not cross or collapse, guaranteeing invertibility.

Reversible networks rely on fixed dimension splitting heuristics, but common splittings interleaved with non-volume conserving elements are constraining and their choice has a big impact on performance [17, 10]. This makes building reversible networks a difficult task, while at the same time, it is unclear how such exotic designs relate to widely used architectures like ResNets [15].

Recently, instead of specifying a discrete architecture, it was proposed to parametrize the derivative of hidden states by a neural network [7, 13]. This approach amounts to learning Lipschitz dynamics of an ordinary differential equation (ODE). While the ODE over a finite time interval is invertible [7], the discretization relies on adaptive ODE solvers to guarantee accurate solutions. This induces computational costs which are hard to control and even get worse during training [13]. In contrast, ResNets have fixed computational cost.

To close this gap, we leverage the viewpoint of ResNets as Euler discretization of ODEs [14, 20, 18, 8] and prove that invertible ResNets can be constructed by simply changing the normalization scheme of standard ResNets. This approach allows unconstrained architectures for each residual block, while only requiring a Lipschitz constant smaller than one for each block. We numerically compute their inverse, which has $\mathcal{O}(1)$ memory cost. Finally, we show that invertible ResNets perform on par with their non-invertible counterparts on classifying MNIST and CIFAR10 images.

2 Unifying ResNets, Invertible Networks and ODEs

The remarkable similarity of ResNets and Euler-methods from ODEs

$$x_{t+1} = x_t + g_{\theta_t}(x_t) = x_t + hf_{\theta_t}(x_t), \tag{1}$$

with activations (states) $x_t \in \mathbb{R}^d$, layer-index (time-step) $t = 1, \ldots, T$, scaling (step-size) $h > 0$ and residual block g_{θ_t} (dynamic of ODE $f_{\theta_t}(x_t)$) has attracted growing research in the intersection of deep learning and dynamical systems [18, 14, 20, 7]. However, little attention has been paid to the dynamics backwards in time

$$x_t = x_{t+1} - hf_{\theta_t}(x_t) = x_{t+1} - g_{\theta_t}(x_t), \tag{2}$$

which amounts to the implicit backward Euler discretization. In particular, solving the dynamics backwards in time would implement an inverse of the corresponding ResNet. As stated in the following theorem, a simple condition suffices to make the dynamics solvable and thus renders the ResNet invertible.

Theorem 1 (Sufficient condition for invertible ResNets). *Let $F_\theta : \mathbb{R}^d \to \mathbb{R}^d$ with $F_\theta = (F_\theta^1 \circ \ldots \circ F_\theta^T)$ denote a ResNet with blocks $F_\theta^t = I + g_{\theta_t}$. Then, the ResNet F_θ is invertible if*

$$\text{Lip}(g_{\theta_t}) < 1, \text{ for all } t = 1, \ldots, T, \tag{3}$$

where $\text{Lip}(g_{\theta_t})$ is the Lipschitz-constant of g_{θ_t}.

Proof. Since ResNet F_θ is a composition of functions, it is invertible if each block F_θ^t is invertible. Let $x_{t+1} \in \mathbb{R}^d$ be arbitrary and consider the backward Euler discretization $x_t = x_{t+1} - hf_{\theta_t}(x_t) = x_{t+1} - g_{\theta_t}(x_t)$. Re-writing as a iteration yields

$$x_t^0 := x_{t+1} \text{ and } x_t^{k+1} := x_{t+1} - g_{\theta_t}(x_t^k), \tag{4}$$

where $\lim_{k \to \infty} x_t^k = x_t$ is the fixed point if the iteration converges. As $g_{\theta_t} : \mathbb{R}^d \to \mathbb{R}^d$ is an operator on a Banach space, the contraction condition $\text{Lip}(g_{\theta_t}) < 1$ guarantees convergence due to the Banach fixed point theorem. $\square$

The condition above was also stated in [2] (Appendix D), however, their proof restricts the domain of the residual block g to be bounded and applies only to linear operators g as the inverse was given by a convergent Neumann-series. Note, that the condition is not necessary, e.g. other approaches [9, 10, 16, 6, 17] rely on triangular structures to create analytical inverses. After proving above condition for invertible ResNets, we discuss connections to related approaches:

Invertibility and ODEs: While ResNets are only guaranteed to be invertible if each residual block is contractive ($\text{Lip}(g_\theta) < 1$), ODEs given by

$$\frac{dx(t)}{dt} = f(x(t), \theta(t)), \text{ for } t \in [0, T] \tag{5}$$

with Lipschitz continuous $f(x(t), \theta(t))$ are reversible [7]. By choosing a scaling $h\,g_\theta$ (time steps h), such that $\text{Lip}(h\,g_\theta) < 1$, bijectivity of the ODE is preserved under the Euler discretization.

Stability of ODE: There are two main approaches to study stability of ODEs, 1) behavior for $t \to \infty$ and 2) Lipschitz stability over finite time intervals $[0, T]$. Based on time-invariant dynamics $f(x(t))$, [8] constructed asymptotically stable ResNets using anti-symmetric layers such that $Re(\lambda(J_x)) < 0$ (with $Re(\lambda(\cdot))$ denoting the real-part of eigenvalues, $\rho(\cdot)$ spectral radius and $J_x g$ the Jacobian at point x). By projecting weights based on the Gershgorin circle theorem, they further fulfilled $\rho(J_x g) < 1$, yielding asymptotically stable ResNets with shared weights over layers.
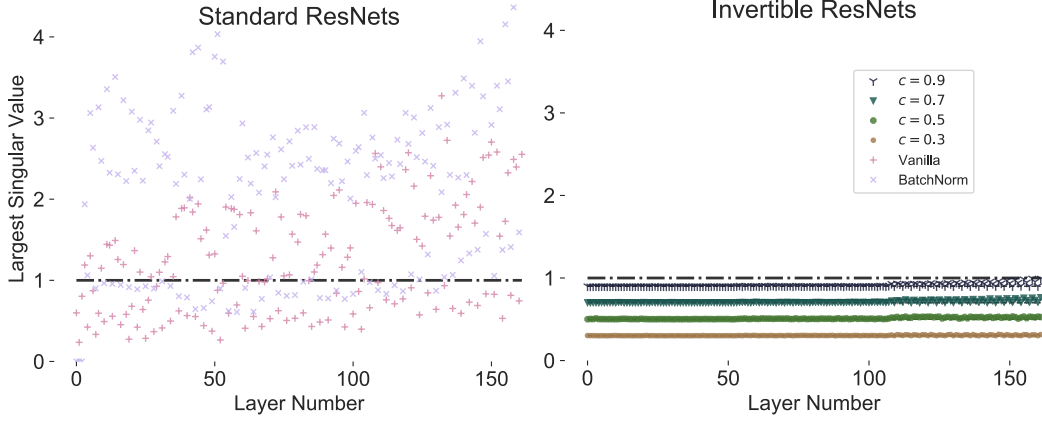
Figure 2: Maximal singular value of each layers convolutional operator for various CIFAR10 trained ResNets. *Left:* Vanilla and Batchnorm ResNet singular values. It is likely that the baseline ResNets are not invertible as roughly two thirds of their layers have singular values fairly above one, making the blocks non-contractive. *Right:* Singular values for our 4 spectrally normalized ResNets. The regularization is effective and in every case the single ResNet block remains a contraction.

On the other hand, [14, 20] considered time-dependent dynamics $f(x(t), \theta(t))$ corresponding to standard ResNets. Similarly, by using anti-symmetric layers they induced stability via leveraging the stability theory of time-invariant ODEs. Yet, the applicability of such an approach to non-linear, time-dependent dynamics is unclear as they only refer to kinematic eigenvalues [5] which generalize the theory of linear time-invariant ODEs to some subclassses (e.g. diagonally dominant matrices, Example 3.7 in [5]) of time-dependent ODEs. Hence, a full understanding of the asymptotic stability of ODEs associated to standard ResNets remains open.

Contrarily, initial value problems on $[0, T]$ are well-posed for Lipschitz continuous dynamics [4]. Thus, the invertible ResNet with $\text{Lip}(f) < 1$ can be understood as a stabilizer of an ODE for step size $h = 1$ without a restriction to anti-symmetric layers as in [20, 14, 8].

Turning discrete into continuous dynamics: The view of deep networks as dynamics over time offers two fundamental learning approaches: 1) Direct learning of continuous ODE dynamics parametrized by neural networks as in [7, 13] and 2) Indirect learning of ODE dynamics using discretization architectures like ResNets [14, 20, 18, 8].

By fixing the ResNet F_θ, the dynamic $f(t, x(t))$ is only fixed at time points t_i, $i = 1, \ldots, T$ corresponding to each block $g_{\theta_{t_i}}$. For example, a linear interpolation in time turns the discretization back in to a continuous set of dynamics. However, in this indirect approach the dynamics depend on the discretization, while the direct approach [7, 13] learns the ODE and adapts the discretization based on the fixed ODE. Thus, the effect of multi-step discretization schemes [18] in the indirect approach is unclear as the nature of the discretization changes the underlying ODE dynamics.

3 Experiments

We show that standard and normalized ResNets perform on par in image classification, while normalized ResNets are guaranteed to be invertible. We train a standard pre-activation ResNet [15] with 55 bottleneck blocks on CIFAR10 and MNIST. All experiments use identical settings for all hyperparameters. We replace subsampling of strided convolution layers with "invertible downsampling" operations [16] to allow invertibility, see appendix A for training and architectural details. To obtain the numerical inverse, we apply 100 fixed point iterations for each block. However, this is just to guarantee full convergence, but in practice much fewer iterations suffice. The trade-off between reconstruction error and number of iterations is analyzed in appendix B.

Satisfying $\text{Lip}(g) \leq c < 1$: We implement residual blocks as $g = W_3 \kappa(W_2 \kappa(W_1))$, where W_i are convolutional layers and κ denotes ReLU. Thus, $\text{Lip}(g) < 1$ if $\|W_i\|_2 < 1$, where $\| \cdot \|_2$ denotes the spectral norm. Unlike [19], we directly estimate the spectral norm of W_i by performing power-iteration using W_i and W_i^T as proposed in [12]. Note, that a power-iteration on the parameter matrix [19] only gives a bound on $\|W_i\|_2$, see [22].

We use only 1 power-iteration during SGD-training, which yields an under-estimate $\tilde{\sigma}_i \geq \|W_i\|_2$.

ResNet:		Vanilla	BN	$c = 0.9$	$c = 0.7$	$c = 0.5$	$c = 0.3$
Classification	CIFAR10	6.95	6.00	6.30	8.23	17.71	24.66
% Error	MNIST	0.61	0.65	0.64	0.61	1.12	1.65
Reconstruction	CIFAR10	1.4	inf	1.3e-7	9.5e-8	1.9e-8	7.2e-9
Normed L2 Error	MNIST	2.5e-7	1.0	2.2e-7	1.9e-7	1.6e-7	7.3e-8
Guaranteed Inverse:		**No**	**No**	**Yes**	**Yes**	**Yes**	**Yes**

Table 1: Classification error on MNIST and CIFAR10, as well as normalized L2 distance between original and reconstructed images on the test set for ResNets with various normalization coefficients.

Then, we normalize as $\tilde{W}_i = c\,W_i/\tilde{\sigma}_i$ if $c/\tilde{\sigma}_i < 1$. Since $\tilde{\sigma}_i$ is an under-estimate, $\|W_i\|_2 \leq c$ is not guaranteed. Thus, after training we compute $\|W_i\|_2$ exactly using the SVD on the Fourier transformed parameter matrix following [21] to show $\mathrm{Lip}(g) < 1$ holds in all cases.

Classification and reconstruction results for two baseline ResNets (with and without BatchNorm) and four invertible ResNets with different spectral normalization coefficients are shown in Table 1. The results illustrate that our proposed invertible ResNets perform on par with the baselines for larger c settings in terms of classification performance, while being provably invertible. When applying very conservative normalization (small c), the classification error becomes higher on both datasets.

The normalized L2 reconstruction errors show, that our regularization is effective and the inverse is close to exact. Intruigingly, our analysis also reveals that ResNets without BatchNorm are invertible after training on MNIST, whereas the BatchNorm ResNet is not. Further, both ResNets with and without BatchNorm are not invertible after training on CIFAR10, as can also be seen from the singular value plots in figure 4.

See figure 3 for qualitative reconstruction results with 100 fixed point iteration steps. Note that the reconstruction error decays quickly and the errors are already imperceptible after 5-20 iterations, which is the cost of 5-20 times the forward pass and empirically corresponds to 0.15-0.75 seconds for reconstructing 100 CIFAR10 images. Computing the inverse is fast even for the largest normalizaton coefficient, but becomes faster with stronger regularization. The iterations needed for full convergence is approximately cut into half when reducing the spectral normalization coefficient by 0.2, see appendix B for a detailed plot.

Figure 3: Original images and reconstructions from vanilla and normalized ResNets with invertible downsampling. Surprisingly, MNIST reconstructions are close to exact for both models, indicating that the vanilla ResNet learns to be invertible without explicitly enforcing it. On CIFAR10 however, reconstructions completely fail for the vanilla ResNet, but are qualitatively and quantitatively exact for our proposed network, without sacrificing classification performance.

In summary, we observe that invertibility without additional constraints is unlikely, but possible, whereas it is hard to predict if networks will have this property. In our proposed normalized ResNets however, we do have the theoretical guarantee of the existence of an inverse without harming classification performance.

Acknowledgements

References

[1] Anonymous. Excessive invariance causes adversarial vulnerability. In *Submitted to International Conference on Learning Representations*, 2019. under review.

[2] Anonymous. Information regularized neural networks. In *Submitted to International Conference on Learning Representations*, 2019. under review.

[3] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W Pellegrini, Ralf S Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. *arXiv preprint arXiv:1808.04730*, 2018.

[4] U.M. Ascher. *Numerical methods for evolutionary differential equations*. Computational science and engineering. Society for Industrial and Applied Mathematics, 2008.

[5] U.M. Ascher, R.M.M. Mattheij, and R.D. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1995.

[6] Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. Reversible architectures for arbitrarily deep residual neural networks. *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[7] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018.

[8] Marco Ciccone, Marco Gallieri, Jonathan Masci, Christian Osendorfer, and Faustino J. Gomez. Nais-net: Stable deep networks from non-autonomous differential equations. *arXiv preprint arXiv:1804.07209*, 2018.

[9] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

[10] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *International Conference on Learning Representations*, 2017.

[11] Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B Grosse. The reversible residual network: Backpropagation without storing activations. *Advances in Neural Information Processing Systems*, 2017.

[12] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael Cree. Regularisation of neural networks by enforcing lipschitz continuity. *arXiv preprint arXiv:1804.04368*, 2018.

[13] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

[14] Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2018.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[16] Jörn-Henrik Jacobsen, Arnold W.M. Smeulders, and Edouard Oyallon. i-revnet: Deep invertible networks. In *International Conference on Learning Representations*, 2018.

[17] Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 2018.

[18] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. *arXiv preprint arXiv:1710.10121*, 2017.

[19] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[20] Lars Ruthotto and Eldad Haber. Deep neural networks motivated by partial differential equations. *arXiv preprint arXiv:1804.04272*, 2018.

[21] Hanie Sedghi, Vineet Gupta, and Philip M. Long. The singular values of convolutional layers. *arXiv preprint arXiv:1805.10408*, 2018.

[22] Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. *Advances in Neural Information Processing Systems*, 2018.

A Experimental Details

We use pre-activation ResNets with 55 convolutional bottleneck blocks with 3 convolution layers each and kernel sizes of 3x3, 1x1, 3x3 respectively. In the BatchNorm version, we apply a batch normalization before every ReLU activation function. The multiplier for the bottleneck is 4. The network has 2 downsampling stages after 18 and 36 blocks each and also concatenates zeros to its input, to increase the number of channels of the input by a factor of 4, a strategy initially described as injective iRevNet [16] and also used in Glow on MNIST [17].

We train for 200 epochs with momentum SGD and a learning rate of 0.1, decayed by a factor of 0.2 after 60, 120 and 160 epochs. Weight decay is set to 5e-4 and a dropout of probability 0.1 is applied in the residual block, which is also taken into account when choosing the normalization coefficients. We apply shifts for MNIST and shifts and flips for CIFAR10 as random data-augmentation during training. The inputs for MNIST are normalized to [-0.5,0.5] and for CIFAR10 as well, after preprocessing each image via subtracting the mean and dividing by the standard deviation of the training set.

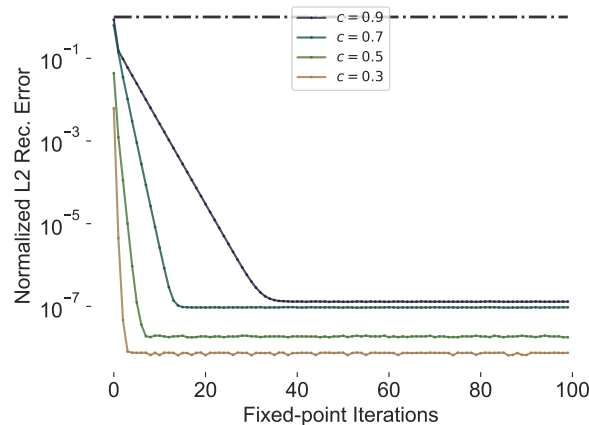## B Fixed Point Iteration Analysis

Figure 4: Tradeoff between number of fixed point iterations and reconstruction error (log scale) for computing the inverse for different normalization coefficients in CIFAR10 trained invertible ResNets. The reconstruction error decays quickly. 5-20 iterations are sufficient respectively to obtain visually perfect reconstructions. Note that one iteration corresponds to the time for one forward pass, thus inversion is approximately 5-20 times slower than inference. This corresponds to a reconstruction time of 0.15-0.75 seconds for a batch of 100 CIFAR10 images with 5-20 iterations and 4.3 seconds with 100 iterations.