Understanding and Mitigating Exploding Inverses in Invertible Neural Networks

Jens Behrmann*[,1], Paul Vicol*[,2,3], Kuan-Chieh Wang*[,2,3], Roger Grosse[2,3], Jörn-Henrik Jacobsen[2,3]
[1]University of Bremen, [2]University of Toronto, [3]Vector Institute

Motivation & Summary

- Invertible neural networks (INNs) have many applications: training generative models w/ exact likelihoods, increasing posterior flexibility in VAEs, computing memory-efficient gradients, solving inverse problems, and analyzing robustness.
- These applications rely on the assumption that theoretical invertibility carries through to the numerical instantiation.
- We show that common INN architectures suffer from exploding inverses & can become numerically non-invertible.
- We provide ways to mitigate this instability: 1) enforcing global stability using Lipschitz-constrained INN architectures or 2) regularization to enforce local stability.

Theory

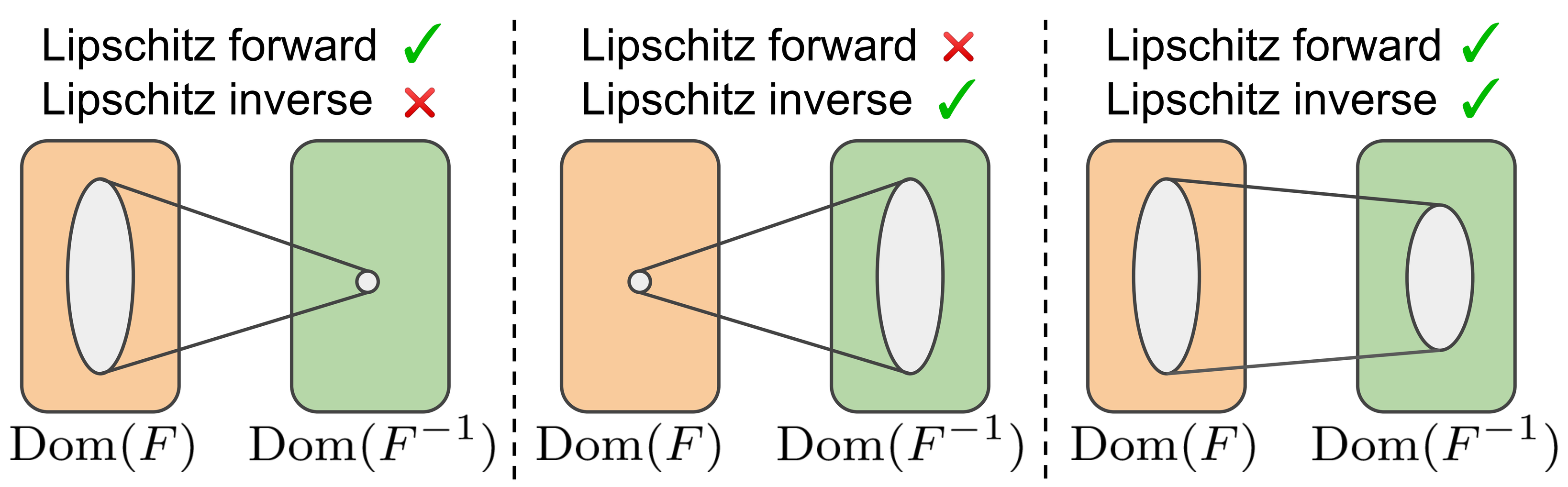
Additive

$F(x)_{I_1} = x_{I_1}$
$F(x)_{I_2} = x_{I_2} + t(x_{I_1})$

Affine

$F(x)_{I_1} = x_{I_1}$
$F(x)_{I_2} = x_{I_2} \odot g(s(x_{I_1})) + t(x_{I_1})$

- Computations are carried out with limited precision → error is always introduced in both the forward and inverse passes.
- Instability in either pass will aggravate this imprecision



- There is a global bound on $\mathrm{Lip}(F)$ and $\mathrm{Lip}(F^{-1})$ for additive blocks, but only local bounds for affine blocks.

Controlling Global Stability

- Additive: Can use spectral norm to control the Lip constant
- Affine: Can increase stability by avoiding scaling by small values. But still no global Lipschitz bound.

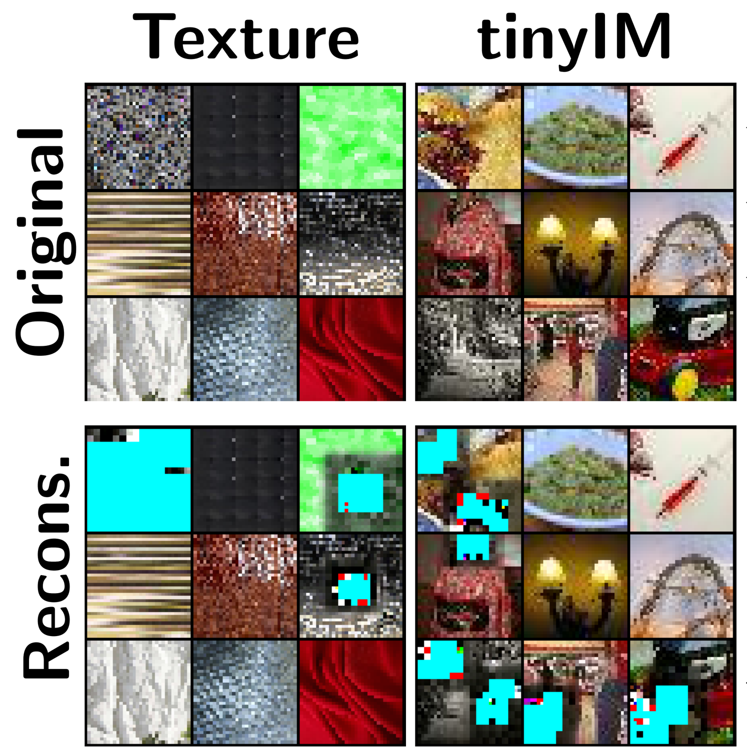
Theory (Contd.)

Controlling Local Stability

- Use penalty on the Jacobian to enforce local stability
  - We propose using an efficient approximation, Bi-Directional Finite Differences Regularization
- Normalizing flow (NF) objective has a stabilizing effect:
  - Prior: pushes output to have small norm, improving forward stability.
  - Log-determinant: increases all singular values, w/ stronger effect on small SVs, improving inverse stability.

INN Instability on OOD Data

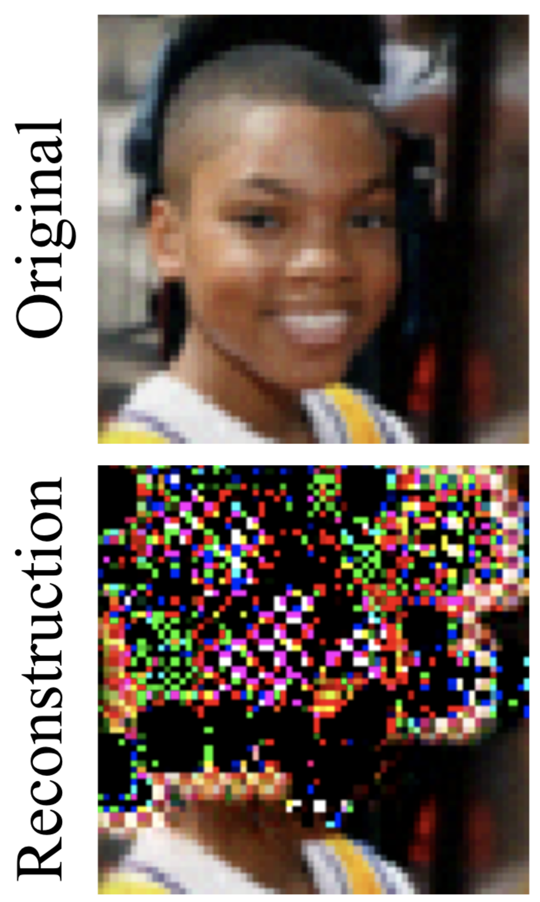
- Global invertibility is needed to apply INNs to OOD data.
- INNs can become numerically non-invertible even when trained with NF (despite encouraging local stability)



	Glow		ResFlow	
Dataset	% Inf	Err	% Inf	Err
CIFAR-10	0	6.3e-5	0	2.9e-2
Uniform	100	-	0	1.7e-2
Gaussian	100	-	0	7.2e-3
SVHN	0	5.5e-5	0	7.3e-2
Texture	37.0	7.8e-2	0	2.0e-2
Places	24.9	9.9e-2	0	2.9e-2
tinyIM	38.9	1.6e-1	0	3.5e-2

- Thus, likelihoods computed by Glow are not meaningful

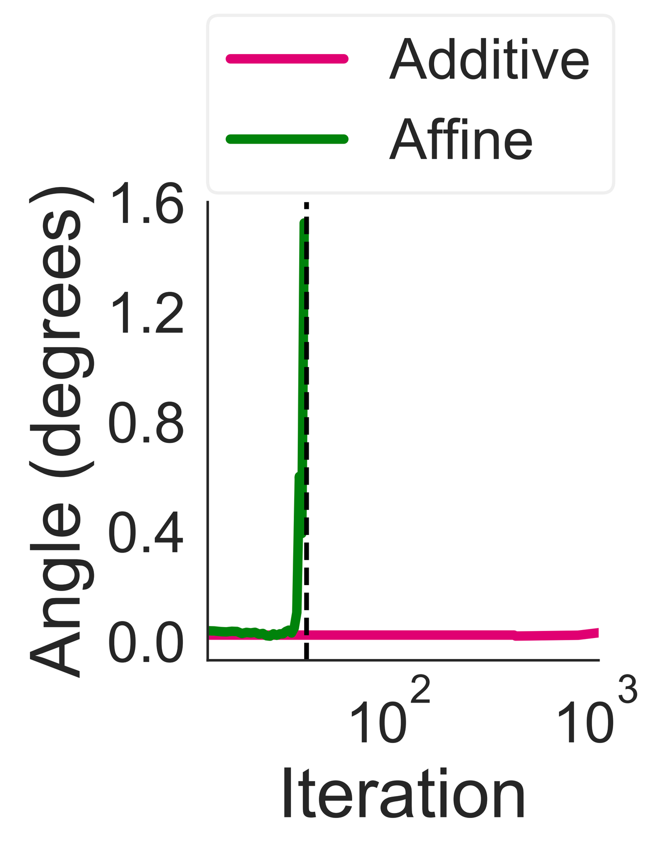
INN Instability in the Data Distribution



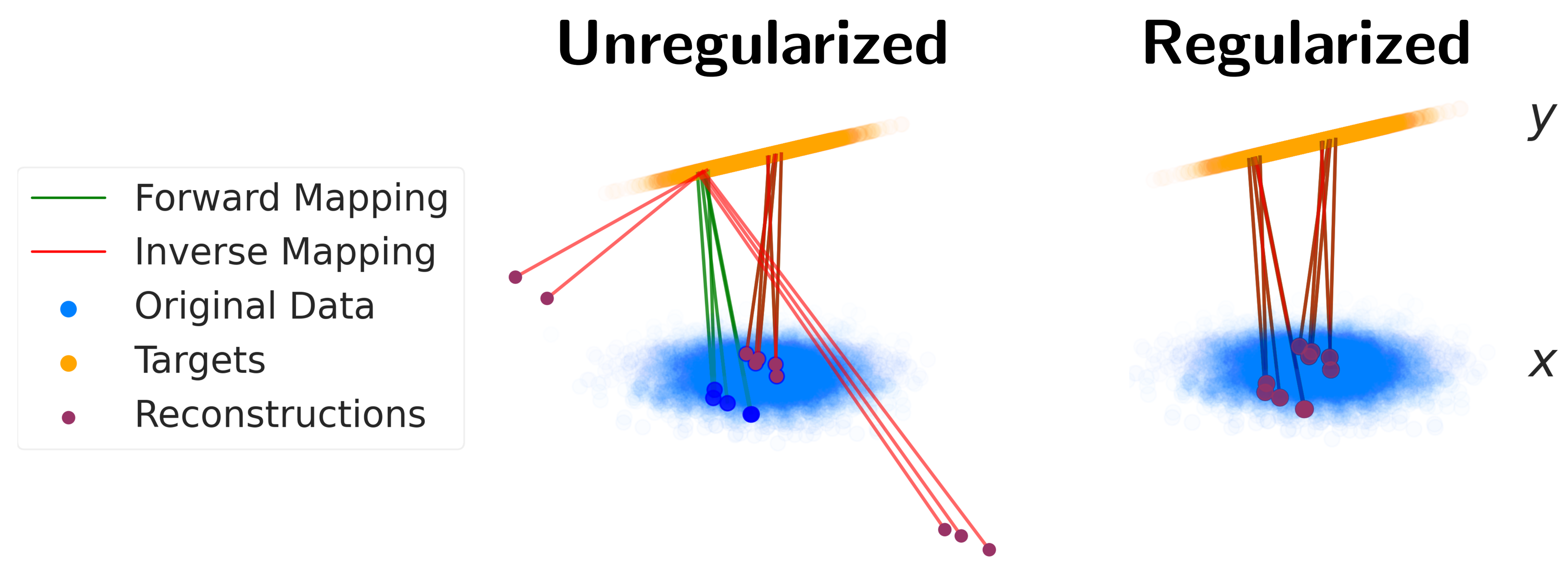
- By optimizing within the dequantization distribution of a datapoint we are able to find regions that are poorly reconstructed by the model.
- Start with x and use Projected Gradient Descent to find a perturbed example x' with high reconstruction error:

$$\arg\max_{||x'-x||_\infty \le \epsilon} ||x' - F^{-1}(F(x'))||_2.$$

Supervised Learning w/ Memory-Efficient Gradients



- INNs enable memory-efficient training by recomputing activations in the backward pass rather than storing them in the forward pass
- Additive and affine INNs achieve similar test accuracy on CIFAR-10, but differ in stability
- While additive is stable, affine gives infinite or nan gradients after a few epochs



- Exploding inverses on a 2D regression task.
- In contrast to NFs, there is no default mechanism to avoid unstable inverses in supervised learning
- Solution: use finite-differences (FD) regularization or add the normalizing flow (NF) objective with small weighting

Model	Reg.	Inv?	Test Acc	Recons. Err.	Min SV	Max SV
Additive	None	✓	89.73	4.3e-2	6.1e-2	4.4e+3
	FD	✓	89.71	1.1e-3	8.7e-2	2.6e+1
	NF	✓	89.52	9.9e-4	3.9e-2	6.6e+1
Affine	None	✗	89.07	Inf	1.9e-12	1.7e+3
	FD	✓	89.47	9.6e-4	9.6e-2	1.5e+1
	NF	✓	89.71	1.3e-3	3.5e-2	7.7e+1

- Instability in the affine model arises from the inverse mapping, as the min SV is 1.9e-12.
- Both FD and NF regularizers stabilize the model without harming accuracy