A. Additional Lemmas and Proofs

Proof. **(Theorem 1)**
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

$$x_t^0 := x_{t+1} \text{ and } x_t^{k+1} := x_{t+1} - g_{\theta_t}(x_t^k), \tag{6}$$

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 $\mathrm{Lip}(g_{\theta_t}) < 1$ guarantees convergence due to the Banach fixed point theorem. □

Remark 5. *The condition above was also stated in Zhao et al. (2019) (Appendix D), however, their proof restricts the domain of the residual block g to be bounded and applies only to linear operators g, because the inverse was given by a convergent Neumann-series.*

Proof. **(Lemma 2)**
First note, that $\mathrm{Lip}(F) \leq 1 + L$ follows directly from the addition of Lipschitz constants. For the inverse, consider

$$\begin{aligned}
\|F(x) - F(y)\|_2 &= \|x - y + g(x) - g(y)\|_2 \\
&= \|x - y - (-g(x) + g(y))\|_2 \\
&\geq |\|x - y\|_2 - \| - g(x) + g(y)\|_2| \\
&\geq |\|x - y\|_2 - \|(-1)\,(g(x) - g(y))\|_2| \\
&\geq \|x - y\|_2 - \|g(x) - g(y)\|_2 \\
&\geq \|x - y\|_2 - L\|x - y\|_2,
\end{aligned} \tag{7}$$

where we apply the reverse triangular inequality in (7) and apply the Lipschitz constant of g. Denote $x = F^{-1}(z)$ and $y = F^{-1}(w)$ for $z, w \in \mathbb{R}^d$, which is possible since F^{-1} is surjective. Inserting above yields

$$\begin{aligned}
&\|F(F^{-1}(z)) - F(F^{-1}(w))\|_2 \geq (1 - L)\|F^{-1}(z) - F^{-1}(w)\|_2 \\
\iff \qquad &\frac{1}{1-L}\|z - w\|_2 \geq \|F^{-1}(z) - F^{-1}(w)\|_2,
\end{aligned}$$

which holds for all z, w. □

Lemma 6 (Positive Determinant of Jacobian of Residual Layer)**.** *Let $F(x) = (I + g(\cdot))(x)$ denote a residual layer and $J_F(x) = I + J_g(x)$ its Jacobian at $x \in \mathbb{R}^d$. If $\mathrm{Lip}(g) < 1$, then it holds λ_i of $J_F(x)$ are positive for all x and thus*

$$|\det[J_F(x)]| = \det[J_F(x)],$$

where λ_i denotes the eigenvalues.

Proof. **(Lemma 6)**
First, we have $\lambda_i(J_F) = \lambda_i(J_g) + 1$ and $\|J_g(x)\|_2 < 1$ for all x due to $\mathrm{Lip}(g) < 1$. Since the spectral radius $\rho(J_g) \leq \|J_g\|_2$, it is $|\lambda_i(J_g)| < 1$. Hence, $Re(\lambda_i(J_F)) > 0$ and thus $\det J_F = \prod_i (\lambda_i(J_g) + 1) > 0$. □

Lemma 7 (Lower and Upper Bounds of an invertible ResNet on log-determinant)**.** *Let $F_\theta : \mathbb{R}^d \to \mathbb{R}^d$ with $F_\theta = (F_\theta^1 \circ \ldots \circ F_\theta^T)$ denote an invertible ResNet with blocks $F_\theta^t = I + g_{\theta_t}$. Then, we can obtain the following bounds*

$$\begin{aligned}
d\sum_{t=1}^{T} \ln(1 - \mathrm{Lip}(g_t)) &\leq \ln|\det J_F(x)| \\
d\sum_{t=1}^{T} \ln(1 + \mathrm{Lip}(g_t)) &\geq \ln|\det J_F(x)|,
\end{aligned}$$

for all $x \in \mathbb{R}^d$.

Proof. **(Lemma 7))**
First, the sum over the layers is due to the function composition, because $J_F(x) = \prod_t J_{F^t}(x)$ and

$$\ln|\det J_F(x)| = \ln\left(\prod_{t=1}^{T}\det J_{F^t}(x)\right) = \sum_{t=1}^{T}\ln\det J_{F^t}(x),$$

where we use the positivity of the determinant, see Lemma 6. Furthermore, note that

$$\sigma_d(A)^d \leq \prod_i \sigma_i(A) = |\det A| \leq \sigma_1(A)^d$$

for a matrix A and largest singular values σ_1 and smallest σ_d. Furthermore, we have $\sigma_i(J_{F^t}) \leq (1 + \mathrm{Lip}(g_t))$ and $\sigma_d(J_{F^t}) \leq (1 - \mathrm{Lip}(g_t))$, which follows from Theorem 2. Inserting this and applying the logarithm rules finally yields the claimed bounds. □

Proof. **(Theorem 3)**
We begin by noting that

$$\begin{aligned}
|PS(J_g, n) - \operatorname{tr}\ln(J_g)| &= \left|\sum_{k=n+1}^{\infty}(-1)^{k+1}\frac{\operatorname{tr}\left(J_g^k\right)}{k}\right| \\
&\leq \sum_{k=n+1}^{\infty}\left|(-1)^{k+1}\frac{\operatorname{tr}\left(J_g^k\right)}{k}\right| \\
&\leq \sum_{k=n+1}^{\infty}\left|\frac{\operatorname{tr}\left(J_g^k\right)}{k}\right| \\
&\leq d\sum_{k=n+1}^{\infty}\frac{\mathrm{Lip}(g)^k}{k}, \qquad (8)
\end{aligned}$$

where inequality (8) follows from

$$\begin{aligned}
|\operatorname{tr}\left(J^k\right)| \leq \left|\sum_{i=d}^{d}\lambda_i(J^k)\right| &\leq \sum_{i=d}^{d}|\lambda_i(J^k)| \leq d\rho(J^k) \\
&\leq d\|J^k\|_2 \leq d\|J\|_2^k \leq d\,\mathrm{Lip}(g)^k. \qquad (9)
\end{aligned}$$

We note that the full series $\sum_{k=1}^{\infty}\frac{\mathrm{Lip}(g)^k}{k} = -\ln(1 - \mathrm{Lip}(g))$ thus we can bound the approximation error by

$$|PS(J_g, n) - \operatorname{tr}\ln(J_g)| \leq -d\left(\ln(1 - \mathrm{Lip}(g)) + \sum_{k=1}^{n}\frac{\mathrm{Lip}(g)^k}{k}\right)$$

□

Proof. **(Theorem 4)**
First, we derive the by differentiating the power series and using the linearity of the trace operator. We obtain

$$\begin{aligned}
\frac{\partial}{\partial\theta_i}\ln\det\left(I + J_g(x,\theta)\right) &= \frac{\partial}{\partial\theta_i}\left(\sum_{k=1}^{\infty}(-1)^{k+1}\frac{\operatorname{tr}\left(J_g^k(x,\theta)\right)}{k}\right) \\
&= \operatorname{tr}\left(\sum_{k=1}^{\infty}\frac{k(-1)^{k+1}}{k}J_g^{k-1}(x,\theta)\frac{\partial(J_g(x,\theta))}{\partial\theta_i}\right) \\
&= \operatorname{tr}\left(\sum_{k=0}^{\infty}(-1)^k J_g^k(x,\theta)\frac{\partial(J_g(x,\theta))}{\partial\theta_i}\right).
\end{aligned}$$

By definition of $\|\cdot\|_\infty$,

$$\|\nabla_\theta PS\big(J_g(\theta),\infty\big) - \nabla_\theta PS\big(J_g(\theta),n\big)\|_\infty = \max_{i=1,\dots,p}\left|\frac{\partial}{\partial\theta_i}PS\big(J_g(\theta),\infty\big) - \frac{\partial}{\partial\theta_i}PS\big(J_g(\theta),n\big)\right|,$$

which is why, we consider an arbitrary i from now on. It is

$$\begin{aligned}\left|\frac{\partial}{\partial\theta_i}PS\big(J_g(\theta),\infty\big) - \frac{\partial}{\partial\theta_i}PS\big(J_g(\theta),n\big)\right| &= \sum_{k=n+1}^{\infty}(-1)^k \operatorname{tr}\left(J_g^k(x,\theta)\frac{\partial(J_g(x,\theta))}{\partial\theta_i}\right)\\ &\leq d\sum_{k=n+1}^{\infty}\operatorname{Lip}(g)^K\left\|\frac{\partial J_g(x,\theta)}{\partial\theta_i}\right\|_2, \end{aligned} \tag{10}$$

where we used the same arguments as in estimation (9).

In order to bound $\left\|\frac{\partial J_g(x,\theta)}{\partial\theta_i}\right\|_2$, we need to look into the design of the residual block. We assume contractive and element-wise activation functions (hence $\phi'(\cdot)<1$) and N linear layers W_i in a residual block. Then, we can write the Jacobian as a matrix product

$$J_g(x,\theta) = W_N^T D_N \cdots W_1^T D_1,$$

where $D_i = diag(\phi'(z_{i-1}))$ with pre-activations $z_{i-1}\in\mathbb{R}^d$.

Since we need to bound the derivative of the Jacobian with respect to weights θ_i, double backpropagation (Drucker & Lecun, 1992) is necessary. In general, the terms $\|W_i^T\|_2$, $\|D_i\|_2$, $\|D_i^*\|_2 := \|diag(\phi''(z_{i-1}))\|_2$, $\left\|\left(\frac{\partial W_i}{\partial\theta_i}\right)\right\|_2$ and $\|x\|_2$ appear in the bound of the derivative. Hence, in order to bound $\left\|\frac{\partial J_g(x,\theta)}{\partial\theta_i}\right\|_2$, we bound the previous terms as follows

$$\|W_i^T\|_2 \leq \operatorname{Lip}(g) \tag{11}$$
$$\|D_i\|_2 \leq const, \tag{12}$$
$$\|D_i^*\|_2 \leq const \tag{13}$$
$$\|x\|_2 \leq const \tag{14}$$
$$\left\|\left(\frac{\partial W_i}{\partial\theta_i}\right)^T\right\|_2 \leq \|W_i\|_F + s. \tag{15}$$

In particular, (12) is due to the assumption of a Lipschitz activation function and (13) due to assuming a Lipschitz derivative of the activation function. Note, that we are using continuously differentiable activations functions (hence, not ReLU), where this assumptions holds for common functions like ELU, softplus and tanh. Furthermore, (14) holds by assuming bounded inputs and due to the network being Lipschitz. To understand the bound (15), we denote s as the amount of parameter sharing of θ_i. For example, if θ_i is a entry from a convolution kernel, $s = w * h$ with w spatial width and h spatial height. Then

$$\left\|\left(\frac{\partial W_i}{\partial\theta_i}\right)^T\right\|_2 \leq \|W_i\|_F + s,$$

since

$$\frac{\partial W_{lm}(x,\theta)}{\partial\theta_i} = \begin{cases}1, & \text{if } W_{lm}=\theta_i\\ 0, & \text{else}\end{cases}.$$

Hence, as each term appearing in the second derivative $\left\|\frac{\partial J_g(x,\theta)}{\partial\theta_i}\right\|_2$ is bounded, we can introduce the constant $a(g,\theta,x)<\infty$ which depends on the parameters, the implementation of g and the inputs x. Note, that we do not give an exact bound on $\left\|\frac{\partial J_g(x,\theta)}{\partial\theta_i}\right\|_2$, since we are only interesting in the existence of such a bound in order to proof the convergence in the claim.

Inserting above in (10) and denoting $c := \mathrm{Lip}(g)$, yields

$$\left| \frac{\partial}{\partial \theta_i} PS\big(J_g(\theta), \infty\big) - \frac{\partial}{\partial \theta_i} PS\big(J_g(\theta), n\big) \right| \leq \; d \; a(g, \theta, x) \sum_{k=n+1}^{\infty} c^K = \tilde{a}(d, g, \theta, x) \left(\frac{1}{1-c} - \left(\frac{1-c^n}{1-c} + c^n \right) \right).$$

Letting $f(n) := \tilde{a}(d, g, \theta, x) \left(\frac{1}{1-c} - \left(\frac{1-c^n}{1-c} + c^n \right) \right)$ and $g(n) = c^n$, then

$$\lim_{n \to \infty} \left| \frac{f(n)}{g(n)} \right| = const < \infty,$$

which proves the claimed convergence rate. □

B. Verification of Invertibility

Inveritibility of Learned Mappings In this experiment we train standard ResNets and i-ResNets with various layer-wise Lipschitz coefficients ($c \in \{.3, .5, .7, .9\}$). After training, we inspect the learned transformations at each layer by computing the largest singular value of each linear mapping based on the approach in Sedghi et al. (2019). It can be seen clearly (Figure 6 left) that the standard and BatchNorm models have many singular values above 1, making their residual connections non-invertible. Conversely, in the i-ResNet models (Figure 6 right), all singular values are below 1 (and roughly equal to c) indicating their residual connections are invertible.

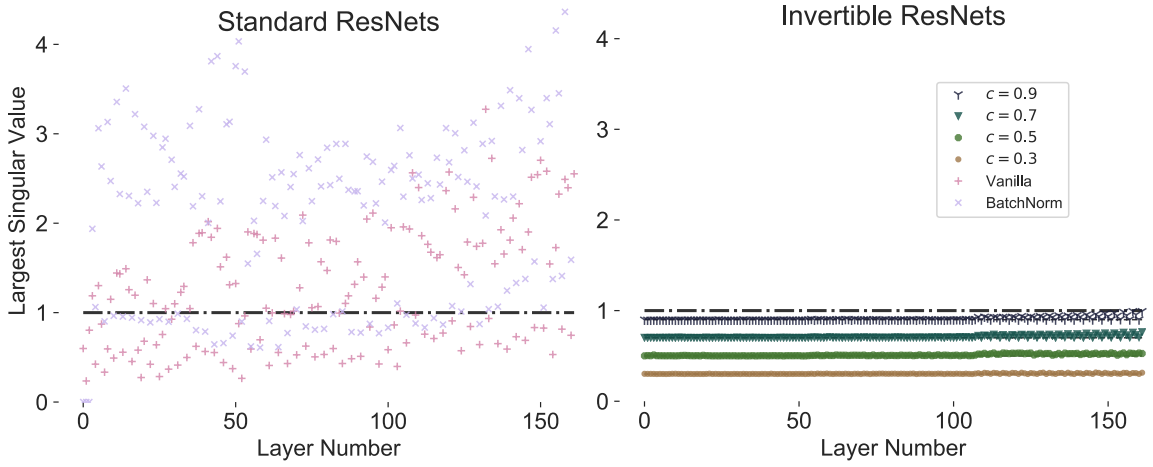

Figure 6. Maximal singular value of each layers convolutional operator for various trained ResNets on Cifar10. *Left:* Vanilla and Batchnorm ResNet singular values. It is likely that the baseline ResNets are not invertible as roughly two thirds of their layers have singular values fairly above one, making 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.

Computing Inverses with Fixed-Point Iteration Here we numerically compute inverses in our trained models using the fixed-point iteration, see Algorithm 1. We invert each residual connection using 100 iterations (to ensure convergence). We see that i-ResNets can be inverted using this method whereas with standard ResNets this is not guaranteed (Figure 7 top). Interestingly, on MNIST we find that standard ResNets are indeed invertible after training on MNIST (Figure 7 bottom).

Figure 7. Original images (top), i-ResNets with $c = 0.9$ (middle) and reconstructions from vanilla (bottom). Surprisingly, MNIST reconstructions are close to exact for both models, even without explicitly enforcing the Lipschitz constant. On CIFAR10 however, reconstructions completely fail for the vanilla ResNet, but are qualitatively and quantitatively exact for our proposed network.

C. Experimental Details

C.1. Runtime Comparison

Glow	i-ResNet	i-ResNet SN	i-ResNet SN LogDet
0.72 sec	0.31 sec	0.57 sec	1.88 sec

Table 5. Timings for a forward and backward pass. Glow is the dimension-splitting baseline, it has the same number of layers and channels as all i-ResNets and batch size is identical. We compare Glow to: 1) plain, 2) spectral norm, 3) spectral norm and log determinant estimated i-ResNets. Discriminative i-ResNets are around 1.2 - 2.1 times faster, while generative i-ResNets are around 2.6 times slower than the dimension splitting baseline. Thus, overall wall-clock times do not differ by much more than a factor of two in all cases.

C.2. Classification

Architecture We use pre-activation ResNets with 39 convolutional bottleneck blocks with 3 convolution layers each and kernel sizes of 3x3, 1x1, 3x3 respectively. All models use the ELU nonlinearity (Clevert et al., 2015). In the BatchNorm version, we apply batch normalization before every nonlinearity and in the invertible models we use ActNorm (Kingma & Dhariwal, 2018) before each residual block. The network has 2 down-sampling stages after 13 and 26 blocks where a dimension squeezing operation is used to decrease the spatial resolution. This reduces the spatial dimension by a factor of two in each direction, while increasing the number of channels by a factor of four. All models transform the data to a 8x8x256 tensor to which we apply BatchNorm, a nonlinearity, and average pooling to a 256-dimensional vector. A linear classifier is used on top of this representation.

Injective Padding Since our invertible models are not able to increase the dimension of their latent representation, we use injective padding (Jacobsen et al., 2018) which concatenates channels of 0's to the input, increasing the size of the transformed tensor. This is analagous to the standard practice of projecting the data into a higher-dimensional space using a non-ResNet convolution at the input of a model, but this mapping is reversible. We add 13 channels of 0's to all models tested, thus the input to our first residual block is a tensor of size 32x32x16. We experimented with removing this step but found it led to approximately a 2% decrease in accuracy for our CIFAR10 models.

Training We train for 200 epochs with momentum SGD and a weight decay of 5e-4. The learning rate is set to 0.1 and decayed by a factor of 0.2 after 60, 120 and 160 epochs. For data-augmentation, we apply random shifts of upt to two pixels for MNIST and shifts/ random horizontal flips for CIFAR(10/100) during training. The inputs for MNIST are normalized to [-0.5,0.5] and for CIFAR(10/100) normalize by subtracting the mean and dividing by the standard deviation of the training set.

C.3. Generative Modeling

Toy Densities We used 100 residual blocks, where each residual connection is a multilayer perceptron with state sizes of 2-64-64-64-2 and ELU nonlinearities (Clevert et al., 2015). We used ActNorm (Kingma & Dhariwal, 2018) after each residual block. The change in log density was computed exactly by constructing the full Jacobian during training and visualization.

MNIST and CIFAR The structure of our generative models closely resembles that of Glow. The model consists of "scale-blocks" which are groups of i-ResNet blocks that operate at different spatial resolutions. After each scale-block, apart from the last, we perform a squeeze operation which decreases the spatial resolution by 2 in each dimension and multiplies the number of channels by 4 (invertible downsampling).

Our MNIST and CIFAR10 models have three scale-blocks. Each scale-block has 32 i-ResNet blocks. Each i-ResNet block consists of three convolutions of $3 \times 3, 1 \times 1, 3 \times 3$ filters with ELU (Clevert et al., 2015) nonlinearities in between. Each convolutional layer has 32 filters in the MNIST model and 512 filters in the CIFAR10 model.

We train for 200 epochs using the Adamax (Kingma & Ba, 2014) optimizer with a learning rate of .003. Throughout training we estimate the log-determinant in Equation (3) using the power-series approximation (Equation (4)) with ten terms for the MNIST model and 5 terms for the CIFAR10 model.

Evaluation During evaluation we use the bound presented in Section 3.3 to determine the number of terms needed to give an estimate with bias less than .0001 bit/dim. We then average over enough samples from Hutchinson's estimator such that the standard error is less than .0001 bit/dim, thus we can safely report our model's bit/dim accurate up to a tolerance of .0002.

Choice of Nonlinearity Differentiating our log-determinant estimator requires us to compute second derivatives of our neural network's output. If we were to use a nonlinearity with discontinuous derivatives (i.e. ReLU), then these values are not defined in certain regions. This can lead to unstable optimization. To guarantee the quantities required for optimization always exist, we recommend using nonlinearities which have continuous derivatives such as ELU (Clevert et al., 2015) or softplus. In all of our experiments we use ELU.

D. Fixed Point Iteration Analysis

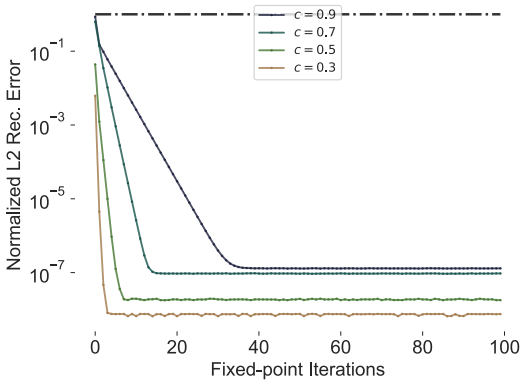

Figure 8. Trade-off between number of fixed point iterations and reconstruction error (log scale) for computing the inverse for different normalization coefficients of trained invertible ResNets (on CIFAR10). The reconstruction error decays quickly. 5-20 iterations are sufficient respectively to obtain 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.

E. Evaluating the Bias of Our Log-determinant Estimator

Here we numerically evaluate the bias of the log-determinant estimator used to train our generative models (Equation (4)). We compare the true value (computed via brute-force) with the estimator's mean and standard deviation as the number of terms in the power series is increased. After 10 terms, the estimator's bias is negligible and after 20 terms it is numerically 0. This is averaged over 1000 test examples.

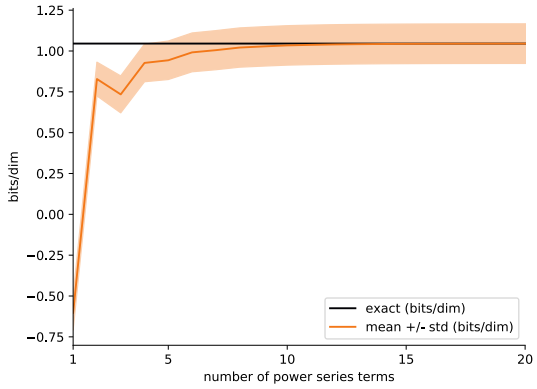

Figure 9. Convergence of approximation error of log-determinant estimator when varying the number of terms used in the power series. The variance is due to the stochastic trace estimator.

F. Additional Samples of i-ResNet flow

CIFAR10 samples.

MNIST samples.