We thank the reviewers for their comments and the largely positive feedback. Reviewers agree that "*the paper clearly touches on very important questions*" (**R5**), since " *efficiently computing (a restricted form of) the Jacobian term*" has been the focus of "*quite a bit of recent research on deep density estimation*" (**R7**). Reviewers also praised the novelty and correctness of the contribution: it is a shared opinion that "*the paper has pointed to an interesting perspective of simplifying the gradient computation*" (**R6**). The improvement our approach provides "*is demonstrated by experiments*" and "*mathematically provable*" (**R5**); "*since it drastically reduces the computation cost of training a fully connected type deep latent variable model, it is certainly a significant contribution*" (**R8**). The contribution was praised as "*elegant*", "*conceptually simple*" (**R7**) and "*potentially applicable in practice*" (**R5**).

A comment which was made (**R5**, **R6**, **R8**) is that, given its wide applicability, it would be interesting to try the method on additional applications. While we agree, we focused on density estimation as it is one of the most challenging problems in probabilistic modeling, and on the theoretical and empirical characterization of the computational improvement, believing (and reviewers seem to agree with us, s.a.) that our work constitutes a significant contribution in these regards.

R6: *Rigorous formulation and convergence properties of relative gradient:* We will add more details on this. Rigorous theory of the relative gradient is well known [Absil et al. , *Optimization Algorithms on Matrix Manifolds*, 2009]; in our paper we propose a simple and accessible derivation. Our proposed method is a stochastic first order optimization algorithm on the manifold of invertible $D \times D$ matrices: almost sure convergence of the parameters to a critical point of the gradient of the cost function can be derived for such SGD with decreasing step size under suitable assumptions (e.g. [Bonnabel, *SGD on Riemannian manifolds*, 2013]). We will include these references in the paper.

R6: *Example problems where normalizing flows suffer compared to relative gradients:* We mention some in the paper, and will further elaborate on this. Most state-of-the-art flow models employ autoregressive transformations and/or coupling layers permuting the inputs between successive layers. These architectures have several limitations, e.g. they can not learn a properly disentangled feature representation. *Linear flows* provide a strict generalization thereof, and our approach can be used for their computationally efficient training. Alternative methods decompose the weight matrix W into easier-to-optimize transformations. One alternative is to compute the PLU decomposition of W and optimize the L and U transformations. The drawback in this approach is that the permutation matrix P cannot be learned. A more flexible alternative is to consider the QR decomposition of W, however computing Q in full generality requires $\mathcal{O}(D^3)$ operations, matching the complexity of the naive optimization of linear flows. An experimental comparison of the performance of the PLU and QR decompositions against the direct optimization of W is in [Hoogeboom et al., *Emerging convolutions for generative normalizing flows*, 2019], describing numerical and stability issues when using PLU. We will include this discussion and reference in the paper.

R6: *Too much emphasis on existing concepts, too little on the proposed approach:* We will try to balance this.
R7: *Computation time in the experiments:* We will add more details to the paper. One epoch on MNIST ($D = 784$, 50k training samples) on a modern laptop CPU takes an order of tens of seconds, a $\sim 4.5\times$ speedup compared to "standard" optimization and $\sim 50\times$ speedup w.r.t. "autodiff" (see below). Our convergence time is ~ 15 min.
Broader impact: We will add a more thorough discussion of this point.
R8: *Experiments use regular SGD, without Adam:* Thanks for pointing this out. We will add experiments with standard SGD. In figure 1 below, we show results on toy datasets like those in figure 2 in the main paper. It can be seen that the data densities are modeled convincingly. We also report (figure 2 below) the evolution of the loss with SGD and Adam on density estimation on MNIST. Optimization with SGD appears to converge slower (confirming the notion that Adam is a more effective optimizer) but ultimately leads to a comparably good result. Similar considerations hold for all datasets in Table 1 in the main paper, with SGD performance being slightly worse but comparable.
R8: *More comments on the projected gradient algorithm:* The augmented matrix formalism allows the computation of the relative gradient for the biases. The projection step corresponds to zeroing out gradients on the last row of the augmented matrix, as remarked by the reviewer. We will report explicit formulas in the appendix.

R6, R8: *In Table 1, what is the number reported? What are the models being compared to?:* The reported numbers are log-likelihoods. The competing models are the same as in reference [34]. We will clarify this in the caption.

R7, R8: *In the main body what is the difference between standard and autodiff in fig. 1?:* "Standard" refers to computing gradients of the Jacobian term as explained in section 3 in the main paper; "autodiff" refers to computing the Jacobian term and its gradients via automatic differentiation with the Jax package. We will clarify this in the main text.

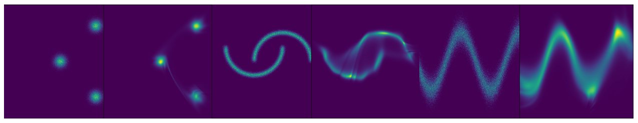
Figure 1: 2D toy examples trained with SGD. True distribution on the left, predicted densities on the right.

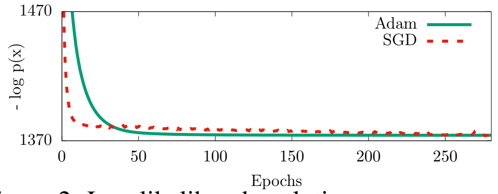

Figure 2: Log-likelihood evolution on MNIST validation set.