Aggregated Responses:

We thank all reviewers for their insightful comments and useful suggestions. Below, we have taken your suggestions which have strengthened the paper.

FID score for sample quality. Reviewers 1 & 3 both suggested reporting FID scores. Our model achieves 46.16 on CIFAR10. For comparison, [2] reports 36.4, 37.11 and 65.93 for WGAN-GP, DCGAN and PixelCNN (lower is better), respectively. We also tested the official Glow model which got a FID score of 46.90, slightly worse than ours.

CelebAHQ 256x256. We have additional experiments on 256x256 images where we achieve 1.00 bits/dim (Glow reports 1.03 bits/dim while using a model double the size of ours). Our samples are similar to Glow's non-temperature annealed samples. However, we note that while Glow used 40 GPUs (with 1 example per GPU and gradient checkpointing), we could train with 3 examples per GPU and a budget of only 4 GPUs.

Reproducibility. We plan on releasing the weights for trained models to allow easier adoption in future works. For instance, this will allow easy computation of various metrics for evaluating sample quality, if needed.

Reviewer #1:

Hybrid modeling experiments. One of the main motivations behind Residual Flows was to create a family of models which is strong in both discriminative and generative tasks. Hybrid density modeling seemed like a natural choice to empirically validate this. We did not provide more downstream experiments (e.g. semi-supervised performance) as we wanted our message to be simple: if a researcher is looking into hybrid models and finding that coupling blocks aren't working well, they should consider residual blocks (with unbiased estimation).

Temperature trick for sampling. We have looked into this and will update the paper. The temperature trick used by Glow only works when the log-determinant does not depend on the sample itself, i.e. additive coupling blocks. This allows an equivalence between reducing entropy in the Gaussian base distribution and a temperature-annealed model. For general flow models, obtaining quality samples efficiently would be a topic on its own.

CelebAHQ 64x64. We used a downsampled version of CelebAHQ which was also qualitatively used by Flow++.

Reviewer #2:

Generalization to other methods/areas. We argue that our contributions bring to light interesting technical innovations that can be generalized to vastly different domains and applications, though we agree that the method of application will likely not be straightforward and require much thought.

Many works that make use of the Russian roulette estimator are already referenced in our related work section, which include graphics and optimization. Furthermore, our estimation approach can lead to improvements in existing works that contain infinite series or log-determinants, such as [3] which approximates the density of GAN samples but may have a bias problem. The gradient formulation is derived using the idea of a Neumann series, which was also used in [1] to derive a "recurrent backpropagation" for training recurrent neural nets.

With regards to applicability within the invertible/flow literature, the design of activation functions with meaningful second-order derivatives will be useful for models that actually compute the log-determinant (or any other objective) using differentiation. Currently, research is still focused on models where the log-determinant is computed using the *output* of a neural net rather than the derivative of one.

Thanks to the reviewer's comments, we may include some of this explanation in the introduction itself.

Reviewer #3:

Inequality in Appendix. Its reason is that the two sides are different infinite series with the same (convergence) value, but different terms. They differ after the unbiased estimators are used. They may have slightly different variances, but we did not notice any significant differences in practice and it's not clear whether one is theoretically always lower variance than the other. We will add more clarifications in the appendix.

Adaptive SN. We always perform power iteration to convergence, so the number of iterations is adaptive. It's usually very high at initialization, then around $1\sim5$ iterations after every weight update, with the mode being 1.

References (We will add [1-2] to the main text. [3] is already cited.)

[1] "Reviving and Improving Recurrent Back-Propagation" Liao et al. (2018)
[2] "Autoregressive Quantile Networks for Generative Modeling" Ostrovski et al. (2018)
[3] "Backpropagation for Implicit Spectral Densities" Ramesh & LeCun. (2018)