We thank the reviewers for their insightful comments. Due to limited space, we respond to your main points below.

[Reviewer 1] 1. The goal of our experiments is to test that, under the same settings, models with Woodbury layers outperform other baselines—not to aim for SOTA bpds. We use smaller models to budget computational resources to test the key hypotheses. As R2 wrote, our experiments fairly compare all models with the same settings. In terms of the bpd improvements, we argue that they are significant enough to demonstrate the improvements of our method. Emerging convolution Hoogeboom et al. [2019] (Table 4) improves 0.02 bpd over 1x1 convolution on ImageNet32. Our experiments show an even more significant improvement: 0.12 bpd on ImageNet32. Finally, as R2 suggested, evaluation with small models is informative and important for low-resource settings, such as mobile devices.

2. We will add expanded experimental details to the appendix. We run each model to fixed number of iterations: 300,000 iterations and around 380 epochs for CIFAR10 and 450,000 iterations and around 20 epochs for ImageNet. We test each method every fixed number of iterations. The bpds in Table 1 are the best bpds obtained by each method. The bpds are single-run results, since each run of the experiment requires 3 to 5 days, and we test 7 models on 3 datasets. So running each model multiple times is a major cost. We tune the hyper-parameters for each model, e.g., weight decay of the optimizer and gradient norm, with a grid search and only running 10,000 iterations, before we run the full training.

3. We will follow your suggestions to move the latent dimension bottleneck experiment to the main paper and list the complexities of methods in a table. Thanks for your ideas for further exploration. We'll add these to our discussion.

[Reviewer 2] 1. We appreciate your positive comments. Let the input be a $c \times h \times w$ tensor. The complexities are $\mathcal{O}(c^2hw + c^3)$ for 1x1 convolutions, $\mathcal{O}(c^2hw)$ for emerging convolutions, and $\mathcal{O}(chw\log(hw) + c^3hw)$ for periodic convolutions. The Woodbury layer complexity is $\mathcal{O}(dchw)$, where d is the latent dimension and can be roughly seen as constant. The complexity of Woodbury layers is comparable to other methods and even smaller than periodic convolutions when c, h, w are big. Therefore, it should be possible to apply Woodbury layers on mobile devices.

[Reviewer 3] 1. We believe that R3 has misunderstood some important points. Woodbury transformations are not meant to be faster than 1x1 convolutions. Instead, they are a low-cost way to model richer interactions among dimensions. Figure 3 shows that Woodbury layers are more efficient than other richer layers in either training or sampling. Figure 5 shows that Woodbury models score better bpd. We will clarify our text to help other readers avoid this confusion.

2. To obtain thorough comparisons with reasonable computational budget, we trained smaller models than those in the original Glow paper. We trained the models long enough to fairly compare the different methods. Kingma and Dhariwal use a very large model and train on 40 GPUs. Our results in Figure 4 use 64x64 images, and Kingma and Dhariwal use 256x256 images. We train each model for 200 epochs, and the curves in Figure 5 suggest that the training converged.

[Reviewer 4] 1. We will be more careful about describing Sylvester flows and explaining how Woodbury flows differ. We did not mean to claim that Sylvester flows cannot be applied to tensors. We meant to say that Berg et al. only (theoretically and empirically) analyze Sylvester's flows on vectors. There is no published application of Sylvester's flows on high-dimensional tensors. So one novelty of Woodbury flows is that they are designed for high-dimensional tensors, with channel transformations and spatial transformations, and two ways, i.e., Woodbury and ME-Woodbury, to combine them. We agree that $z' = z + Ah(Bz + b)$ defines a very general set of flows, but one needs to define efficient methods to use specific instantiations of it. Berg et al. analyze one special variant of it, with $A = QR$, and $B = \tilde{R}Q^T$. Woodbury flows is another variant that allows tractable inversion. These are the key novelties of Woodbury flows when compared to the flows analyzed in Berg et al..

2. Thank you for pointing out another way to compute FFT. We report the FFT complexity for the method described by Hoogeboom et al. [2019] (footnote 2). In our code, we directly use the PyTorch FFT implementation. We agree a better FFT algorithm can theoretically make the periodic convolutions faster, and we already mentioned it in the paper (line 239). We will make this caveat clearer when discussing the comparision to periodic convolution.

3. About the running time comparisons, these are measurements of running time for models used in the other experiments to provide a view of the relationship between running time and modeling power. The methods all scale differently with parameter size, but we did not empirically test that because it is easily analyzed theoretically based on the complexities of each method's matrix operations. It's more important to compare the cost of methods that are tuned to best model the data. The ME variant only improves memory storage, and it does not save running time because it still has a bottleneck of a matrix product the same size as the full Woodbury. (See Appendix B for details.)

4. About the local structure, the squeeze layers in flows cause true convolution operations to no longer have the same spatial interpretation as in traditional CNNs. So no current flow architectures have nice spatial interpretation.

5. We did not claim to achieve or target SOTA bpds. We compare with modern flow layers under the same settings, and Woodbury transformations outperform them. More discussions are in our response to Reviewer 1.