1. Training details

In this section, we give more details about the training procedure. As mentioned in the main paper, we used Adam [3] as optimizer in all of our experiments. We pre-trained the denoiser with N2N loss (Eq. 5 of the main paper) for 2,000 epochs. Also note that the denoiser pre-training step was used only to boost training under different setups, and is not a vital part of the overall training. Training the original Noise2NoiseFlow model from scratch will also produce almost the same results ($NLL: -3.498$, $D_{KL}: 0.0275$, PSNR: 52.65).

The supervised DnCNN was trained with MSE using the clean/noisy pairs from SIDD-Medium. Both denoiser pretraining and supervised training used an initial learning rate of $10^{-3}$, which was decayed to $10^{-4}$ at epoch 30, and $5 \times 10^{-5}$ at epoch 60. We used orthogonal weight initialization [2] for the denoiser architectures and the exact same initial weights for the noise model as used in the Noise Flow paper.

The denoiser was a 9 layer DnCNN and was the same in all experiments except where noted. Noise Flow was re-implemented in PyTorch [4] and carefully tested for consistency against the original implementation. Joint training used a constant learning rate of $10^{-4}$ for 2,000 epochs though no improvements were generally observed after $\sim 600$ epochs.

2. Synthetic Noise Experiment

In order to demonstrate that our framework can retrieve the parameters of a supervised trained noise model, we have conducted a synthetic noise experiment. In this setting, we first trained a heteroscedastic Gaussian noise model, which was implemented as a flow layer in Noise Flow. For simplicity, we only took one camera and one ISO setting—namely, iPhone 7 and 800 as ISO level as we had adequate image data for training and evaluation. Under the mentioned setting, the model only has two trainable parameters—namely, $\beta_1$ and $\beta_2$. We then use this trained model to synthesize noisy image pairs for training a subsequent Noise2NoiseFlow model from scratch with only a heteroscedastic Gaussian layer as its noise model and DnCNN as its denoiser. The results shown in Figure 1 shows that our model can successfully retrieve the parameters of a trained NLF model.

3. Failure Cases

Although no significant unrealistic behaviour was noticed, we visualize 5 noise samples with the worst $D_{KL}$ for Noise2NoiseFlow in Figure 2. While the noise samples are not in the best alignment with the real samples, the generated noise patches do not look very unnatural.

References


*Work performed while interns at the Samsung AI Center–Toronto.
Figure 2. Noise synthesis samples from (a) the AWGN model, (b) Camera NLF, (c) Calibrated P-G [5], (d) Noise Flow [1], and our proposed method, Noise2NoiseFlow, compared to the (f) real noise in SIDD for patches where Noise2NoiseFlow has the worst $D_{KL}$ numbers.

