GAN2GAN: Generative Noise Learning for Blind Denoising with Single Noisy Images

Sungmin Cha [1] **Taeeon Park** [1] **Byeongjoon Kim** [2] **Jongduk baek** [2] **and Taesup Moon** [1 3]

Abstract

We tackle a challenging blind image denoising problem, in which only single distinct noisy images are available for training a denoiser, and no information about noise is known, except for it being zero-mean, additive, and independent of the clean image. In such a setting, which often occurs in practice, it is not possible to train a denoiser with the standard discriminative training or with the recently developed Noise2Noise (N2N) training; the former requires the underlying clean image for the given noisy image, and the latter requires two independently realized noisy image pair for a clean image. To that end, we propose GAN2GAN (Generated-Artificial-Noise to Generated-Artificial-Noise) method that first learns a generative model that can 1) simulate the noise in the given noisy images and 2) generate a rough, noisy estimates of the clean images, then 3) iteratively trains a denoiser with subsequently synthesized noisy image pairs (as in N2N), obtained from the generative model. In results, we show the denoiser trained with our GAN2GAN achieves an impressive denoising performance on both synthetic and real-world datasets for the blind denoising setting.

1. Introduction

Image denoising is one of the oldest problems in image processing and low-level computer vision, yet it still attracts lots of attention due to the fundamental nature of the problem. A vast number of algorithms have been proposed over the past several decades, and recently, the CNN-based methods, *e.g.*, (Cha & Moon, 2019) (Zhang et al., 2017) (Tai et al., 2017) (Liu et al., 2018), became the throne-holders in terms of the PSNR performance. The main approach of the most CNN-based denoisers is to apply the discriminative learning framework with (clean, noisy) image pairs and *known* noise distribution assumption. While being effective, such framework also possesses a couple of limitations that become critical in practice; the assumed noise distribution may be mismatched to the actual noise in the data or obtaining the noise-free clean target images is not always possible or very expensive, *e.g.*, medical imaging (CT or MRI) or astrophotographs. The more additional related works are explained in Supplementary Material (S.M.)

In this paper, we consider the pure *unsupervised* blind denoising setting where only *single* distinct noisy images are available for training. Namely, nothing is known about the noise other than it being zero-mean, additive, and independent of the clean image, and neither the clean target images for blind training nor the noisy image pairs for N2N training is available. The crux of our method is to first learn a Wasserstein GAN (Arjovsky et al., 2017)-based generative model that can 1) learn and simulate the noise in the given noisy images and 2) generate rough, initially denoised images. Using such generative model, we then synthesize noisy image pairs by corrupting each of the initially denoised images with the simulated noise *twice* and use them to train a CNN denoiser as in the N2N training (*i.e.*, Noisy N2N). We further show that *iterative* N2N training with refined denoised images can significantly improve the final denoising performance. We dubbed our method as GAN2GAN (Generated-Artifical-Noise to Generated-Artificial-Noise) and show that the denoiser trained with our method can achieve (sometimes, even outperform) the performance of the standard supervised-trained or N2N-trained blind denoisers for the white Gaussian noise case. Furthermore, for mixture/correlated noise or real-world noise in microscopy/CT images, for which the exact distributions are hard to know *a priori*, we show our denoiser significantly outperforms those standard blind denoisers.

2. Motivation

In order to develop the core intuition for motivating our method, we first consider a simple, single-letter Gaussian noise setting. Let $Z = X + N$ be the noisy observation

[1]Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Korea [2]School of Integrated Technology and Yonsei Institute of Convergence Technology, Yonsei University, Incheon, Korea [3]Department of Artifical Intelligence, Sungkyunkwan University, Suwon, Korea. Correspondence to: Taesup Moon <tsmoon@skku.edu>.

of $X \sim \mathcal{N}(0, \sigma_X^2)$, corrupted by the $N \sim \mathcal{N}(0, \sigma_N^2)$. It is well known that the minimum MSE (MMSE) estimator of X given Z is $f^*_{\text{MMSE}}(Z) = \mathbb{E}(X|Z) = \frac{\sigma_X^2}{\sigma_X^2+\sigma_N^2} Z$. We now identify the optimality of N2N in this setting.

N2N Assume that we have two i.i.d. copies of the noise N: N_1 and N_2. Then, let $Z_1 = X + N_1$ and $Z_2 = X + N_2$ be the two independent noisy observation pairs of X. The N2N in this setting corresponds to obtaining the MMSE estimator of Z_2 given Z_1,

$$f_{\text{N2N}}(Z_1) \triangleq \arg\min_f \mathbb{E}(Z_2 - f(Z_1))^2 = \mathbb{E}(Z_2|Z_1)$$
$$= \mathbb{E}(X + N_2|Z_1) \overset{(a)}{=} \mathbb{E}(X|Z_1) = \frac{\sigma_X^2}{\sigma_X^2 + \sigma_N^2} Z_1, \quad (1)$$

in which (a) follows from N_2 being independent of Z_1. Note (1) has the exact same form as $f^*_{\text{MMSE}}(Z)$, hence, estimating X with $f_{\text{N2N}}(Z)$ also achieves the MMSE, in line with (Lehtinen et al., 2018).

"Noisy" N2N Now, consider the case in which we again have the two i.i.d. N_1 and N_2, but the noisy observations are of a *noisy* version of X. Namely, let $X' = X + N_0$, in which $N_0 \sim \mathcal{N}(0, \sigma_0^2)$, and denote $Z_1' = X' + N_1$ and $Z_2' = X' + N_2$ as the noisy observation pairs. Then, we can define a "Noisy" N2N estimator as the MMSE estimator of Z_2' given Z_1',

$$f_{\text{Noisy N2N}}(Z_1', y) \triangleq \arg\min_f \mathbb{E}(Z_2' - f(Z_1'))^2$$
$$= \mathbb{E}(X'|Z_1') = \frac{\sigma_X^2(1+y)}{\sigma_X^2(1+y) + \sigma_N^2} Z_1', \quad (2)$$

in which we denote $y \triangleq \sigma_0^2/\sigma_X^2$ and assume that $0 \le y < 1$. Note clearly (2) coincides with (1) when $y = \sigma_0^2 = 0$. Following N2N, (2) is essentially estimating X' based on $Z' = X' + N$. An interesting subtle question is what happens when we use the mapping $f_{\text{Noisy N2N}}(Z, y)$ for estimating X given $Z = X + N$, *not* X' given Z'. Our theorem below, of which proof is in the Supplementary Material (S.M.), shows that for a sufficiently large σ_0^2, $f_{\text{Noisy N2N}}(Z, y)$ gives a better estimate of X than X'.

Theorem 1 *Consider the single-letter Gaussian setting and $f_{\text{Noisy N2N}}(Z, y)$ obtained in (2). Also, assume $0 < y < 1$. Then, there exists some y_0 s.t. $\forall y \in (y_0, 1)$, $\mathbb{E}(X - f_{\text{Noisy N2N}}(Z, y))^2 < \sigma_0^2$.*

Theorem 1 provides a simple, but useful, intuition that motivates our method; if simulating the noise in the images is possible, we may carry out the N2N training iteratively, provided that a rough *noisy* estimate of the clean image is initially available. Namely, we can first simulate the noise to generate noisy observation pairs of the initial noisy estimate, then do the Noisy N2N training with them to obtain a denoiser that may result in a better estimate of the clean image when applied to the actual noisy image subject to denoising (as in Theorem 1). Then, we can refine the estimates by *iterating* the Noisy N2N training with the generated noisy observation pairs of the previous step's estimate of the clean image, until convergence. To check whether above intuition is valid, we carry out a feasibility experiment and the experimental result is proposed in S.M.

3. Main Method: Three Components of GAN2GAN

To concretely describe our method, we first set the notations. We assume the noisy image $\mathbf{Z}$ is generated by $\mathbf{Z} = \mathbf{x} + \mathbf{N}$, in which $\mathbf{x}$ denotes the underlying clean image and $\mathbf{N}$ denotes the zero-mean, additive noise that is independent of $\mathbf{x}$. For training a denoiser, we do not assume either the distribution or the covariance of $\mathbf{N}$ is known. Moreover, we assume only a database of n *distinct* noisy images, $\mathcal{D} = \{\mathbf{Z}^{(i)}\}_{i=1}^n$, is available for learning a denoiser. A CNN-based denoiser is denoted as $\hat{\mathbf{X}}_{\boldsymbol{\phi}}(\mathbf{Z})$ with $\boldsymbol{\phi}$ being the model parameter, and we use the standard quality metrics, PSNR/SSIM, for evaluation. Our method consists of three parts; 1) smooth noisy patch extraction, 2) training a generative model, and 3) iterative GAN2GAN training of $\hat{\mathbf{X}}_{\boldsymbol{\phi}}(\mathbf{Z})$, each of which we elaborate below.

3.1. Smooth noisy patch extraction

The first step is to extract the noisy image patches from $\mathcal{D}$ that correspond to smooth, homogeneous areas. Our extraction method is similar to that of the GCBD proposed in (Chen et al., 2018), but we make a critical improvement.

The GCBD determines a patch $\boldsymbol{p}$ (of pre-determined size) is smooth if it satisfies the following for *all* of its smaller sub-patches, $\mathbf{q}_j$, with some hyperparameters $\mu, \gamma \in (0, 1)$:

$$|\mathbb{E}(\mathbf{q}_j) - \mathbb{E}(\boldsymbol{p})| \le \mu\mathbb{E}(\boldsymbol{p}), \;\; |\mathbb{V}(\mathbf{q}_j) - \mathbb{V}(\boldsymbol{p})| \le \gamma\mathbb{V}(\boldsymbol{p}), \quad (3)$$

in which $\mathbb{E}(\cdot)$ and $\mathbb{V}(\cdot)$ are the empirical mean and variance of the pixel values.

While (3) works for extracting smooth patches to some extent, it does not rule out choosing patches with high-frequency repeating patterns, which are far from being smooth. Thus, we instead use the 2D discrete wavelet transform (DWT) for a new extraction rule; namely, we determine $\boldsymbol{p}$ is smooth if its four sub-band decompositions obtained by DWT, $\{W_k(\boldsymbol{p})\}_{k=1}^4$, satisfy

$$\frac{1}{4}\sum_{k=1}^{4} \Big| \hat{\boldsymbol{\sigma}}(W_k(\boldsymbol{p})) - \mathbb{E}[\hat{\boldsymbol{\sigma}}_W(\boldsymbol{p})] \Big| \le \lambda\mathbb{E}[\hat{\boldsymbol{\sigma}}_W(\boldsymbol{p})], \quad (4)$$

in which $\hat{\boldsymbol{\sigma}}(\cdot)$ is the empirical standard deviation of the wavelet coefficients, $\mathbb{E}[\hat{\boldsymbol{\sigma}}_W(\boldsymbol{p})] \triangleq \frac{1}{4}\sum_{k=1}^4 \hat{\boldsymbol{\sigma}}(W_k(\boldsymbol{p}))$, and $\lambda \in (0, 1)$ is a hyperparameter. This rule is much simpler than (3), which has to be evaluated for all the sub-patches, $\{\mathbf{q}_j\}$. Once N patches are extracted from $\mathcal{D}$ using (4), we subtract each patch with its mean pixel value, and obtain a set of "noise" patches, $\mathcal{N} = \{\boldsymbol{n}^{(j)}\}_{j=1}^N$. Such subtraction is valid since all the pixel values should be close to their mean in a smooth patch, and the noise is assumed to be zero-mean, additive.

3.2. Training a W-GAN based generative model

Equipped with $\mathcal{D} = \{\mathbf{Z}^{(i)}\}_{i=1}^n$ and the extracted noise patches $\mathcal{N} = \{\boldsymbol{n}^{(j)}\}_{j=1}^N$, we train a generative model, which can learn and simulate the noise as well as generate initial noisy estimates of the clean images, hence, realize the Noisy N2N training explained in Section 2. As shown in Figure 2 (in S.M.), our model has three generators, $\{g_{\boldsymbol{\theta}_1}, g_{\boldsymbol{\theta}_2}, g_{\boldsymbol{\theta}_3}\}$, and two critics, $\{f_{\boldsymbol{w}_1}, f_{\boldsymbol{w}_2}\}$, in which the subscripts stand for the model parameters. The loss functions associated with the components of our model are:

$$\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{\theta}_1, \boldsymbol{w}_1) \triangleq \mathbb{E}_{\boldsymbol{n}}\big[f_{\boldsymbol{w}_1}(\boldsymbol{n})\big] - \mathbb{E}_{\boldsymbol{r}}\big[f_{\boldsymbol{w}_1}(g_{\boldsymbol{\theta}_1}(\boldsymbol{r}))\big] \quad (5)$$

$$\mathcal{L}_{\mathbf{Z}}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{w}_2) \triangleq \mathbb{E}_{\mathbf{Z}}\big[f_{\boldsymbol{w}_2}(\mathbf{Z})\big] - \mathbb{E}_{\mathbf{Z},\boldsymbol{r}}\big[f_{\boldsymbol{w}_2}(g_{\boldsymbol{\theta}_2}(\mathbf{Z}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r}))\big] \quad (6)$$

$$\mathcal{L}_{\text{cyc}}(\boldsymbol{\theta}_2, \boldsymbol{\theta}_3) \triangleq \mathbb{E}_{\mathbf{Z}}\big[\|\boldsymbol{z} - g_{\boldsymbol{\theta}_3}(g_{\boldsymbol{\theta}_2}(\mathbf{Z}))\|_1\big]. \quad (7)$$

The loss (5) is a standard W-GAN (Arjovsky et al., 2017) loss for training the first generator-critic pair, $(g_{\boldsymbol{\theta}_1}, f_{\boldsymbol{w}_1})$, of which $g_{\boldsymbol{\theta}_1}$ learns to generate the independent realization of the noise mimicking the patches in $\mathcal{N} = \{\boldsymbol{n}^{(j)}\}_{j=1}^N$, taking the random vector $\boldsymbol{r} \sim \mathcal{N}(0, I)$ as input. The second loss (6) links the two generators, $g_{\boldsymbol{\theta}_1}$ and $g_{\boldsymbol{\theta}_2}$, with the second critic, $f_{\boldsymbol{w}_2}$. The second generator $g_{\boldsymbol{\theta}_2}$ is intended to generate the *estimate* of the underlying clean patch for $\mathbf{Z}$, *i.e.*, coarsely denoise $\mathbf{Z}$, and the critic $f_{\boldsymbol{w}_2}$ determines how close the distribution of the *generated* noisy image, $g_{\boldsymbol{\theta}_2}(\mathbf{Z}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r})$, is to the that of $\mathbf{Z}$[1]. Our intuition is, if $g_{\boldsymbol{\theta}_1}$ can realistically simulate the noise, then enforcing $g_{\boldsymbol{\theta}_2}(\mathbf{Z}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r})$ to mimick $\mathbf{Z}$ would result in learning a reasonable initial *denoiser* $g_{\boldsymbol{\theta}_2}$. One important detail regarding $g_{\boldsymbol{\theta}_2}$ is its final activation *must* be the sigmoid function for stable training. The third loss (7), which resembles the cycle loss in (Zhu et al., 2017), imposes the encoder-decoder structure between $g_{\boldsymbol{\theta}_2}$ and $g_{\boldsymbol{\theta}_3}$, hence, helps $g_{\boldsymbol{\theta}_2}$ to compress the most redundant part of $\mathbf{Z}$, *i.e.*, the noise, and carry out the initial denoising. Once the losses are defined, training the generators and critics are done in an alternating manner, as in the training of W-GAN (Arjovsky et al., 2017), to approximately solve

$$\min_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\boldsymbol{\theta}_3} \max_{\boldsymbol{w}_1,\boldsymbol{w}_2} \Big[\alpha\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{\theta}_1, \boldsymbol{w}_1) + \beta\mathcal{L}_{\mathbf{Z}}(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{w}_2) + \gamma\mathcal{L}_{\text{cyc}}(\boldsymbol{\theta}_2, \boldsymbol{\theta}_3)\Big], \quad (8)$$

in which (α, β, γ) are hyperparameters to control the trade-offs between the loss functions. There are a couple of subtle points for training with the overall objective (8), and we describe the full details on model architectures, hyperparameters, and training procedures in the S.M.

3.3. Iterative GAN2GAN training of a denoiser

With our generative model, we then carry out the iterative Noisy N2N training as described in Section 2, with the *generated* noisy images. Namely, given each $\mathbf{Z}^{(i)} \in \mathcal{D}$, we generate the pair

$$(\hat{\mathbf{Z}}_{11}^{(i)}, \hat{\mathbf{Z}}_{12}^{(i)}) \triangleq (g_{\boldsymbol{\theta}_2}(\mathbf{Z}^{(i)}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r}_{11}^{(i)}), g_{\boldsymbol{\theta}_2}(\mathbf{Z}^{(i)}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r}_{12}^{(i)})), \quad (9)$$

in which $\boldsymbol{r}_{11}^{(i)}, \boldsymbol{r}_{12}^{(i)} \in \mathbb{R}^{128}$ are i.i.d. $\sim \mathcal{N}(\mathbf{0}, I)$. In contrast to the ideal case in Section 2, each generated image in (9) is a noise-corrupted version of $g_{\boldsymbol{\theta}_2}(\mathbf{Z}^{(i)})$, in which the corruption is done by the *simulated* noise $g_{\boldsymbol{\theta}_1}(\boldsymbol{r})$. Denoting the set of such pairs as $\hat{\mathcal{D}}_1 = \{(\hat{\mathbf{Z}}_{11}^{(i)}, \hat{\mathbf{Z}}_{12}^{(i)})\}_{i=1}^n$, a denoiser $\hat{X}_{\boldsymbol{\phi}}(\mathbf{Z})$ is trained by minimizing $\mathcal{L}_{\text{G2G}}(\boldsymbol{\phi}, \hat{\mathcal{D}}_1) \triangleq \frac{1}{n}\sum_{i=1}^n(\hat{\mathbf{Z}}_{11}^{(i)} - \hat{X}_{\boldsymbol{\phi}}(\hat{\mathbf{Z}}_{12}^{(i)}))^2$. In $\mathcal{L}_{\text{G2G}}(\cdot)$, we only use the generated noisy images and do *not* use the actual observed $\mathbf{Z}^{(i)}$, hence, we dubbed our training as GAN2GAN (G2G) training. Now, denoting the learned denoiser as G2G$_1$ (with parameter $\boldsymbol{\phi}_1$), we can iterate the G2G training. For the j-th iteration (with $j \geq 2$), we generate

$$\big(\hat{\mathbf{Z}}_{j1}^{(i)}, \hat{\mathbf{Z}}_{j2}^{(i)}\big) \triangleq \big(\hat{X}_{\boldsymbol{\phi}_{j-1}}(\mathbf{Z}^{(i)}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r}_{j1}^{(i)}), \hat{X}_{\boldsymbol{\phi}_{j-1}}(\mathbf{Z}^{(i)}) + g_{\boldsymbol{\theta}_1}(\boldsymbol{r}_{j2}^{(i)})\big), \quad (10)$$

for each $\mathbf{Z}^{(i)}$ and denote the resulting set of the pairs as $\hat{\mathcal{D}}_j$. Note in (10), we *update* the noisy estimate of the clean image with the output of G2G$_{j-1}$. Then, the new denoiser G2G$_j$ is obtained by computing $\boldsymbol{\phi}_j \triangleq \arg\min_{\boldsymbol{\phi}} \mathcal{L}_{\text{G2G}}(\boldsymbol{\phi}, \hat{\mathcal{D}}_j)$, where the minimization is done via warm-starting from $\boldsymbol{\phi}_{j-1}$.

4. Experimental results

4.1. Data and experimental settings

Data & training details In synthetic noise experiments, we always used the noisy training images from BSD400 (BSD) (Martin et al., 2001). For evaluation, we used the standard BSD68 (Roth & Black, 2009) as a test set. For real-noise experiment, we experimented on two data sets: the WF set in the microscopy image datasets in (Zhang et al., 2019) and the reconstructed CT dataset. For both datasets, we trained/tested on each (Avg = n) and each dose level, respectively, which corresponds to different noise levels. The details on the experimental settings and baslines is proposed in S.M.

4.2. Denoising results on synthetic noise

White Gaussian noise Table 1 shows the results on BSD68 corrupted by white Gaussian noise with different σ's. Several variations of our G2G, $g_{\boldsymbol{\theta}_2}$ and the G2G iterates, G2G$_{j\geq1}$, are shown for two different training data versions

[1]We assume $g_{\boldsymbol{\theta}_2}$ implicitly has the cropping step for $\mathbf{Z}$ such that the dimension of $g_{\boldsymbol{\theta}_2}(\mathbf{Z})$ and $g_{\boldsymbol{\theta}_1}(\boldsymbol{r})$ match.

Table 1. Results on *BSD68/Gaussian*. Boldface denotes algorithms that only use single noisy images. Red and blue denotes the highest and second highest result among those algorithms, respectively.

PSNR/SSIM	Baselines				G2G variation				Upper Bound
	BM3D	DnCNN-B	N2N	N2V	g_{θ_2}	G2G$_1$	G2G$_2$	G2G$_3$	N2C(Eq.(4))
$\sigma = 15$	**31.07/0.8717**	31.44/0.8836	31.20/0.8745	**29.48/0.8199**	**25.94/0.7519**	**30.98/0.8552**	**32.51/0.8827**	**31.45/0.8825**	31.64/0.8870
$\sigma = 25$	**28.56/0.8013**	28.92/0.8137	28.74/0.8041	**26.97/0.7083**	**24.16/0.6630**	**28.23/0.7669**	**28.82/0.8056**	**28.96/0.8080**	29.11/0.8189
$\sigma = 30$	**27.78/0.7727**	28.06/0.7812	27.91/0.7720	**26.38/0.6657**	**23.43/0.5967**	**27.58/0.7413**	**27.99/0.7783**	**28.03/0.7759**	28.28/0.7890
$\sigma = 50$	**25.60/0.6866**	25.78/0.6721	25.71/0.6712	**24.30/0.5765**	**20.58/0.4482**	**25.08/0.6215**	**25.55/0.6639**	**25.78/0.6749**	26.03/0.6951

Table 2. Results on *BSD68/Mixture & Correlated noise*. The boldface and colored texts are as before.

PSNR/SSIM			Baselines				G2G variation				Upper bound
			BM3D	DnCNN-B	N2N	N2V	g_{θ_2}	G2G$_1$	G2G$_2$	G2G$_3$	N2C(Eq.(4))
Mixture noise	Case A	$s = 15$	**41.44/0.9822**	39.62/0.9749	40.59/0.9860	**33.53/0.9368**	**31.85/0.9522**	**42.35/0.9876**	**42.56/0.9888**	**42.49/0.9885**	42.92/0.9843
		$s = 25$	**37.97/0.9647**	37.23/0.9616	37.39/0.9737	**31.62/0.9057**	**32.73/0.9478**	**39.13/0.9761**	**39.64/0.9800**	**39.72/0.9807**	40.42/0.9843
	Case B	$s = 30$	**30.12/0.8549**	30.58/0.8655	30.58/0.8655	**28.10/0.7543**	**27.55/0.7728**	**29.05/0.8199**	**30.32/0.8456**	**30.49/0.8538**	30.78/0.8685
		$s = 50$	**29.27/0.8190**	30.20/0.8547	30.20/0.8547	**28.22/0.7755**	**27.36/0.7712**	**29.78/0.8345**	**30.04/0.8392**	**30.00/0.8417**	30.39/0.8574
Correlated noise		$\sigma = 15$	**29.84/0.8504**	30.84/0.9011	30.69/0.9223	**28.80/0.8367**	**28.13/0.8370**	**30.73/0.8889**	**31.09/0.8949**	**31.26/0.8954**	31.60/0.9075
		$\sigma = 25$	**26.69/0.7544**	27.39/0.8257	27.32/0.8594	**26.11/0.7348**	**25.68/0.7607**	**27.80/0.8130**	**28.01/0.8271**	**28.00/0.8447**	28.42/0.8376

for learning the generative model.

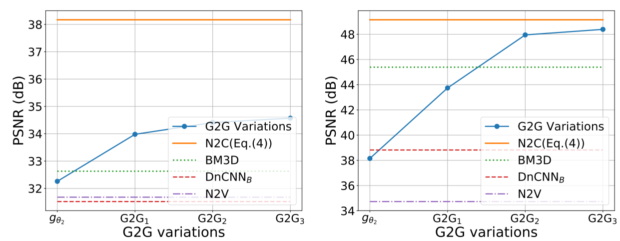
Firstly, we clearly observe the iterative G2G training is *very* effective; namely, it significantly improves the initial noisy estimate $g_{\boldsymbol{\theta}_2}$, particularly when the quality of the initial estimate is not good enough. Secondly, we note G2G$_1$ already considerably outperforms N2V (Krull et al., 2019), which is trained with the *exact* same model architecture and dataset. Finally, the performance of G2G$_3$ is *very* strong; it outperforms BM3D (Dabov et al., 2007), which knows true σ, and even sometimes outperforms the blindly trained DnCNN-B (Zhang et al., 2017) and N2N (Lehtinen et al., 2018), which is trained with the same BSD400 dataset, but with more information. This somewhat counter-intuitive result is possible since our G2G$_j$ accurately learns the correct noise level in the image, while DnCNN-B and N2N are trained with the composite noise levels, $\sigma \in [0, 55]$.

Mixture and correlated noise Table 2 shows the results on mixture and correlated noise beyond white Gaussian. Note our G2G$_j$ does not assume any distributional or correlation structure of the noise, hence, it can still run as long as the assumption on the noise holds. In the table, the G2G results are for (BSD) as specified above. Moreover, DnCNN-B and N2N are still blindly trained with the *mismatched* white Gaussian noise.

From the table, we first note that DnCNN-B and N2N suffer from serious performance degradation for both mixture and correlated noises due to noise mismatch, and the conventional BM3D outperforms them for some cases (*e.g.*, Case A for mixture noise). However, we note our G2G$_2$ can still denoise very well after just two iterations and outperforms all the baselines for all noise types, solely based on single noisy images. Note N2V seriously suffers and is *not* comparable to ours. Finally, N2C(Eq.(4)) is a sound upper bound for all noise types, confirming the correctness of the extraction rule (4).

4.3. Denoising results on real noise

We also test our method on the real-world noise. While some popular real noise is known to have source-dependent characteristics, there are also cases in which the noise is source-independent and pixel-wise correlated, which satisfies the assumption of our method. We tested on two such datasets, the Wide-Focal (WF) set in the microscopy image dataset (Zhang et al., 2019) and a Reconstructed CT dataset. A more detailed description and analysis on these two datasets are in S.M. The WF and Reconstructed CT data has 5 sets (Avg = $1, 2, 4, 8, 16$) and 4 sets (Dose=$25, 50, 75, 100$) with different noise levels, respectively. We did *not* exploit the fact that the images are multiple noisy measurements of a clean image, which enables employing N2N, but treated them as noisy images of distinct clean images. Figure 1(a) and 1(b) shows the PSNR of all methods for each dataset, respectively, averaged over all sets. The baselines were DnCNN-B and BM3D and N2V. We note BM3D estimated noise σ using the method in (Chen et al., 2015). We iterated until G2G$_3$ and N2C(Eq.(4)) was an upper bound for each set. We clearly observe that the performance of G2G$_j$ significantly improves (over $g_{\boldsymbol{\theta}_2}$) as the iteration continues. In results, G2G$_3$ becomes significantly better than DnCNN-B and N2V as well as BM3D, still one of the strongest baselines for real-world noise denoising when no clean target images are available, for both datasets. We report more detailed experimental results (including SSIM) on both datasets in S.M. Moreover, the inference time for BM3D is about 4.5∼5.0 seconds per image since a noise estimation has to be done for each image separately, whereas that for G2G$_j$ is only 4 ms (on GPU), which is another significant advantage of our method. The visualizations on the denoising result are shown in the S.M.



(a) Wide-Focal (WF) (b) Reconstructed CT

Figure 1. Results on real microscopy image denoising on WF and medical image denoising.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017.

Cha, S. and Moon, T. Fully convolutional pixel adaptive image denoiser. In *International Conference on Computer Vision (ICCV)*, 2019.

Chen, G., Zhu, F., and Ann Heng, P. An efficient statistical method for image noise level estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 477–485, 2015.

Chen, J., Chen, J., Chao, H., and Yang, M. Image blind denoising with generative adversarial network based noise modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. Image Processing*, 16(8):2080–2095, 2007.

Krull, A., Buchholz, T.-O., and Jug, F. Noise2void-learning denoising from single noisy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., and Aila, T. Noise2noise: Learning image restoration without clean data. In *International Conference on Machine Learning (ICML)*, 2018.

Liu, D., Wen, B., Fan, Y., Loy, C. C., and Huang, T. S. Non-local recurrent network for image restoration. In *Neural Information Processing Systems (NeurIPS)*, 2018.

Martin, D., Fowlkes, C., Tal, D., and Malik, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision (ICCV)*, 2001.

Roth, S. and Black, M. Field of experts. *International Journal of Computer Vision*, 82(2):205–229, 2009.

Tai, Y., Yang, J., Liu, X., and Xu, C. Memnet: A persistent memory network for image restoration. In *International Conference on Computer Vision (ICCV)*, 2017.

Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Processing*, 26(7):3142 – 3155, 2017.

Zhang, Y., Zhu, Y., Nichols, E., Wang, Q., Zhang, S., Smith, C., and Howard, S. A poisson-gaussian denoising dataset with real fluorescence microscopy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision (ICCV)*, pp. 2223–2232, 2017.

Supplementary Materials for GAN2GAN: Generative Noise Learning for Blind Denoising with Single Noisy Images

Sungmin Cha [1] **Taeeon Park** [1] **Byeongjoon Kim** [2] **Jongduk baek** [2] **and Taesup Moon** [1 3]

1. Related work

Several works have been proposed to overcome the limitation of the vanilla supervised learning based denoising. As mentioned above, Noise2Self (N2S) (Batson & Royer, 2019) and Noise2Void (N2V) (Krull et al., 2019) recently applied self-supervised learning (SSL) approach to train a denoiser only with single noisy images. Their settings exactly coincide with ours, but we show later that our GAN2GAN significantly outperforms them. More recently, (Laine et al., 2019) improved N2V by incorporating specific noise likelihood models with Bayesian framework, however, their method *required* to know the exact noise model and could not be applied to more general, unknown noise settings. Similarly, (Soltanayev & Chun, 2018) proposed SURE (Stein's Unbiased Risk Estimator)-based denoiser that can also be trained with single noisy images, but it worked only with the *Gaussian* noise. Their work was extended in (Zhussip et al., 2019), but it required noisy image *pairs* as in N2N as well as the Gaussian noise constraint.

Table 1. Summary of different settings among the recent baselines.

Alg.\ Requirements	Clean image	Noisy "pairs"	Noise model
N2N (Lehtinen et al., 2018)	✗	✓	✗
HQ SSL (Laine et al., 2019)	✗	✗	✓
SURE (Soltanayev & Chun, 2018)	✗	✗	✓
Ext. SURE (Zhussip et al., 2019)	✗	✓	✓
GCBD (Chen et al., 2018)	✓	✗	✗
N2V (Krull et al., 2019)	✗	✗	✗
GAN2GAN (Ours)	✗	✗	✗

(Chen et al., 2018) devised GCBD method to learn and generate noise in the given noisy images using W-GAN (Arjovsky et al., 2017) and utilized the unpaired clean images to build a supervised training set. Our GAN2GAN is related to (Chen et al., 2018), but we significantly improve their noise learning step and do *not* use the clean data at all. Table 1 summarizes and compares the settings among the above mentioned recent baselines. We clearly see that only our GAN2GAN and N2V do not utilize any "sidekicks" that other methods take advantage of.

More classical denoising methods typically are capable of denoising solely based on the single noisy images by applying various principles, *e.g.*, filtering-based (Buades et al., 2005; Dabov et al., 2007), optimization-based (Elad & Aharon, 2006; Mairal et al., 2009), Wavelet-based (Donoho & Johnstone, 1995), and effective prior-based (Zoran & Weiss, 2011). Those methods typically are, however, computationally intensive during the inference time and cannot be *trained* from a separate set of noisy images, which limits their denoising performance. Another line of recent work worth mentioning is the deep learning-based priors or regularizers, *e.g.*, (Ulyanov et al., 2018; Yeh et al., 2018; Lunz et al., 2018), but their PSNRs still fell short of the supervised trained CNN-based denoisers.

[1]Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Korea [2]School of Integrated Technology and Yonsei Institute of Convergence Technology, Yonsei University, Incheon, Korea [3]Department of Artifical Intelligence, Sungkyunkwan University, Suwon, Korea. Correspondence to: Taesup Moon <tsmoon@skku.edu>.

2. Proof of Theorem 1

Theorem 1 *Consider the single-letter Gaussian setting and $f_{\text{Noisy N2N}}(Z, y)$ obtained in (Eq.(2), manuscript). Also, assume $0 < y = \sigma_0^2/\sigma_X^2 < 1$. Then, there exists some y_0 s.t. $\forall y \in (y_0, 1)$, $\mathbb{E}(X - f_{\text{Noisy N2N}}(Z, y))^2 < \sigma_0^2$.*

Proof: We consider the following chain of equalities:

$$\begin{aligned} & \sigma_0^2 - \mathbb{E}(X - f_{\text{Noisy N2N}}(Z,y))^2 && (1)\\ = \; & \sigma_0^2 - \mathbb{E}\Big(X - \frac{\sigma_X^2(1+y)}{\sigma_X^2(1+y)+\sigma_N^2}(X+N)\Big)^2 && (2)\\ = \; & \sigma_0^2 - \mathbb{E}\Big(\frac{\sigma_N^2}{\sigma_X^2(1+y)+\sigma_N^2}X - \frac{\sigma_X^2(1+y)}{\sigma_X^2(1+y)+\sigma_N^2}N\Big)^2 && (3)\\ = \; & \sigma_X^2\Big(y - \frac{\sigma_N^4}{(\sigma_X^2(1+y)+\sigma_N^2)^2} - \frac{\sigma_N^2\sigma_X^2(1+y)^2}{(\sigma_X^2(1+y)+\sigma_N^2)^2}\Big) && (4)\\ = \; & \sigma_X^2\frac{y(\sigma_X^2(1+y)+\sigma_N^2)^2 - \sigma_N^4 - \sigma_N^2\sigma_X^2(1+y)^2}{(\sigma_X^2(1+y)+\sigma_N^2)^2} && (5) \end{aligned}$$

Now, by denoting the numerator of (5) as $g(y)$, we have

$$\begin{aligned} g(y) &= y\big(\sigma_X^4(1+y)^2 + \sigma_N^4 + 2\sigma_X^2\sigma_N^2(1+y)\big) - \sigma_N^2\sigma_X^2(y^2+2y+1) - \sigma_N^4 && (6)\\ &= \sigma_X^4 y^3 + (2\sigma_X^4 + \sigma_N^2\sigma_X^2)y^2 + (\sigma_X^4 + \sigma_N^4)y - \sigma_N^2\sigma_X^2 - \sigma_N^4. && (7) \end{aligned}$$

Then, we can easily see that

$$\begin{aligned} g(0) &= -\sigma_N^2\sigma_X^2 - \sigma_N^4 < 0 && (8)\\ g(1) &= 4\sigma_X^4 > 0 && (9)\\ g(y) &= 3\sigma_X^4 y^2 + 2(2\sigma_X^4 + \sigma_N^2\sigma_X^2)y + \sigma_X^4 + \sigma_N^4 > 0. && (10) \end{aligned}$$

Therefore, in $0 < y < 1$, we can see $g(y)$ is an increasing function and has a root y_0 in the interval. Hence, the claim of the theorem: for all $y \in (y_0, 1)$, $\mathbb{E}(X - f_{\text{Noisy N2N}}(Z,y))^2 < \sigma_0^2$ holds. ■

3. The feasibility experiment on NoisyN2N

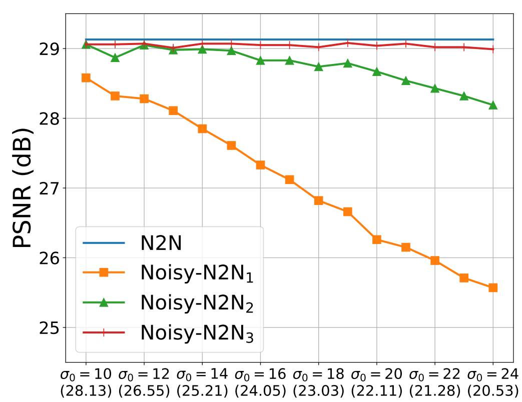
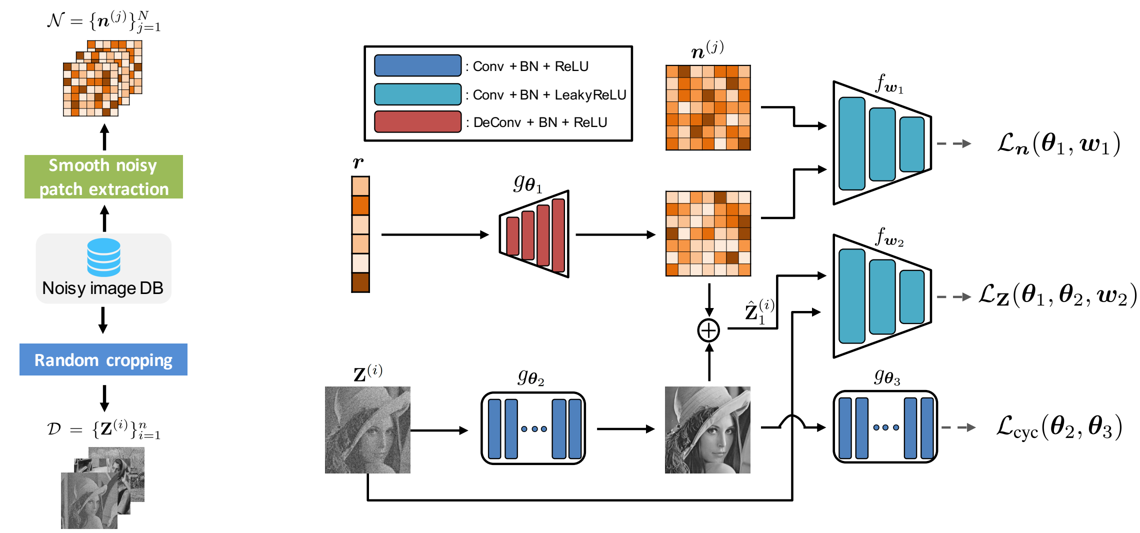


Figure 1. Iterative Noisy N2N.

Figure 1 shows the denoising results on BSD68 (Roth & Black, 2009) for Gaussian noise with $\sigma = 25$. The blue line is the PSNR of the N2N model trained with noisy observation pairs of the *clean* images in the BSD training set, serving as an upper bound. The orange line, in contrast, is the PSNR of the Noisy N2N$_1$ model that is trained with the noisy observation pairs of the *noisy* estimates for the clean images, which were set to be another Gaussian noise-corrupted training images. The standard deviations (σ_0) of the Gaussian for generating the noisy estimates are given in the horizontal axis, and the corresponding PSNRs of the estimates are given in the parentheses. Although Noisy N2N$_1$ clearly lies much lower than the

N2N upper bound, we note its PSNR is still higher than that of the initial noisy estimates, which is in line with Theorem 1. Now, if we iterate the Noisy N2N with the previous step's denoised images (*i.e.*, Noisy-N2N$_2$/Noisy-N2N$_3$ for second/third iterations, respectively), we observe that the PSNR significantly improves and approaches the ordinary N2N for most of the initial σ_0 values. Thus, we observe the intuition from Theorem 1 generalizes well to the image denoising case in an ideal setting, where the noise can be perfectly simulated, and the initial noisy estimates are Gaussian corrupted versions. The remaining question is whether we can also obtain similar results for the blind image denoising setting. We show our generative model-based approach in details in the next section.

4. Overall structure of the W-GAN based generative model



(a) Getting $\mathcal{D}$ & $\mathcal{N}$ (b) The model architecture with three generators and two critics.

Figure 2. Overall structure of the W-GAN based generative model.

5. Details on the experimental settings

5.1. Training details

For the generative model training, the patch size used for $\mathcal{D}$ and $\mathcal{N}$ was 96×96, and n and N were set to $20,000$ (BSD) and $40,000$ (microscopy), respectively. For the iterative G2G training, the patch size for $\mathcal{D}$ was 120×120 and $n = 20,500$, and in every mini-batch, we generated new noisy pairs with $g_{\boldsymbol{\theta}_1}$ as in the noise augmentation of (Zhang et al., 2017). The architecture of G2G$_j$ was set to 17-layer DnCNN in (Zhang et al., 2017).

5.2. Baselines

The baselines were BM3D (Dabov et al., 2007), DnCNN-B (Zhang et al., 2018), N2N (Lehtinen et al., 2018), and N2V (Krull et al., 2019). We reproduced and trained DnCNN-B, N2N and N2V using the publicly available source codes on the *exactly* same training data as our iterative G2G training. For DnCNN-B and N2N, which use either clean targets or two independent noisy image copies, we used 20-layers DnCNN model with composite additive white Gaussian noise with $\sigma \in [0, 55]$. N2V considers the same setting as ours and uses the *exact* same architecture as G2G$_j$; more details on N2V are also given in the S.M. We could not compare with the scheme in (Laine et al., 2019), since their code cannot run beyond white Gaussian noise case in our experiments and they had an unfair advantage: they *newly* generate noisy images by corrupting given clean images for *every* mini-batch whereas we assume the given noisy images are fixed once for all. It is known that such noise augmentation significantly can increase the performance, and their code could not run in our setting in which the noisy images are fixed once given. As an upper bound, we implemented N2C(Eq.(4)), denoting a 17-layer DnCNN trained with clean target images in BSD400 and their noisy counterpart, which is corrupted by our $g_{\boldsymbol{\theta}_1}$ learned with Eq.(4) in the manuscript.

5.3. Details on Mixture noise

For mixture noise, we tested with two cases. Case A corresponds to the same setting as given in (Chen et al., 2018), *i.e.*, 70% $\sim \mathcal{N}(0, 0.1^2)$, 20% $\sim \mathcal{N}(0, 1)$, and 10% $\sim$ Unif$[-s, s]$ with $s = 15, 25$. For case B, we tested with larger variances, *i.e.*, 70% Gaussian $N(0, 15^2)$, 20% Gaussian $N(0, 25^2)$, and 10% Uniform $[-s, s]$ with $s = 30, 50$.

5.4. Details on Correlated noise

For correlated noise, we generated a spatially correlated Gaussian noise, of which neighboring $k \times k$ blocks are correlated and marginal standard deviation $\sigma = 15, 25$. For this, we generated the following noise for each ℓ-th pixel,

$$N_\ell = \eta M_\ell + (1-\eta)\Big(\frac{1}{\sqrt{|\mathcal{NB}_\ell|}} \sum_{m \in \mathcal{NB}_\ell} M_m\Big), \ \ \ell = 1, 2, \dots$$

in which $\{M_\ell\}$ are white Gaussian $\mathcal{N}(0, \sigma^2)$, $\mathcal{NB}_\ell$ is the $k \times k$ neighborhood patch except for the pixel ℓ, and η is a mixture parameter. We set $\eta = 1/\sqrt{2}$ such that the marginal distribution of N_ℓ is also $\mathcal{N}(0, \sigma^2)$ and set $k = 16$. Note in this case, N_ℓ has a spatial correlation, and we tested with $\sigma = 15, 25$.

5.5. Smooth noisy patch extraction

5.5.1. GCBD (Chen et al., 2018) rule Eq.(3)

The original GCBD paper (Chen et al., 2018) did not provide any source code or training data, hence, we reproduced their noisy patch extraction algorithm. There are six hyperparameters for the rule [Eq.(3), Manuscript], and we used the exact same hyperparameters given in their paper, which are shown in Table 2. d and h denote the size of a patch, $\boldsymbol{p}$, and its sub-patches, $\mathbf{q}_j$, given in [Eq.(3), Manuscript], respectively. s_p and s_q are the stride sizes for extracting the patches, $\boldsymbol{p}$ and $\{\mathbf{q}_j\}$, from a given image. μ and λ are the hyperparameters of the rule for selecting the smooth patches shown in [Eq.(3), Manuscript].

Table 2. Hyperparameters for the patch extraction rule of GCBD

Hyperparameters	d	h	s_p	s_q	μ	γ
Values	64	16	32	16	0.1	0.25

5.5.2. G2G rule Eq.(4)

There are three hyperparemeters for our extraction rule [Eq.(4), Manuscript], λ, d (the patch size), and s_d (the stride size for extracting patches from an image). The choices for our experiments are shown in Table 4. Moreover, we stress that we did *not* tune λ using clean images, but the different λ values in the table are determined by the pre-determined number of extracted patches by applying our rule [Eq.(4), Manuscript]. Moreover, as argued in Section 3.1 (manuscript), we do not require any sub-patches to be extracted, hence, have only half the hyperparameters compared to the GCBD rule.

Table 3. Hyperparameters for the extraction rule of G2G

	Gaussian Noise	Mixture Noise	Correlated Noise	WF	Medical
λ	0.03	0.1	0.15	0.42	0.015
d	96				
s_p	24				

5.5.3. Effect of λ

Table 4 shows the effect of λ in [Eq.(4), Manuscript] on the final performance of G2G$_2$. Note the smaller the λ, the less number of patches are extracted, but the homogeneity increases. The table shows λ clearly affects the denoising performance of $g_{\boldsymbol{\theta}_2}$, but as the iterative G2G training continues, the performance of G2G$_2$ becomes not very sensitive to λ.

Hence, in our experiments, we did not optimize λ based on *any* clean validation set, but just set λ based on the number of extracted patches and checking the visual qualities of the patches.

Table 4. Effects of varying λ on the denoising performance.

Gaussian Noise ($\sigma = 25$)				Mixture Noise ($s = 25$)				Correlated Gaussian Noise ($\sigma = 25$)			
λ	# of patches	g_{θ_2}	G2G$_2$	λ	# of patches	g_{θ_2}	G2G$_2$	λ	# of patches	g_{θ_2}	G2G$_2$
0.03	100,000	26.30/0.7123	**28.93/0.8293**	0.1	80,000	32.73/0.9478	**40.30/0.9845**	0.15	100,000	25.68/0.7606	**27.85/0.8185**
0.01	61,000	27.20/0.7159	**28.84/0.8045**	0.005	45,000	35.84/0.9588	**40.16/0.9838**	0.11	30,000	26.61/0.7566	**27.67/0.8203**
0.0075	32,000	26.44/0.7085	**28.80/0.8060**	0.025	23,000	34.20/0.9398	**40.28/0.9848**	0.1	11,000	26.30/0.7440	**27.65/0.8203**

5.6. Training a W-GAN based generative model

Here, we elaborate a couple of subtle points for training our generative model as mentioned in Section 3.2 (manuscript).

Firstly, given the overall optimization objective [Eq.(8), manuscript], we use $(\alpha, \beta, \gamma) = (1, 1, 0)$ for the inner maximization for critics, and use $(\alpha, \beta, \gamma) = (5, 1, 10)$ for the outer minimization for generators. The main intuition for using different (α, β, γ) for training the generators is due to different levels of confidence in the generator loss terms. Namely, we assign the largest weight to [Eq.(7), Manuscript] since it is a deterministic loss and its value has a clear meaning. The generator loss [Eq.(5), Manuscript], which is in the form of the standard W-GAN loss, gets the medium level weight since the meaning of its value is less certain than [Eq.(6), Manuscript]. In contrast, the generator loss in [Eq.(5), Manuscript], which consists of two generators, can become somewhat unstable during training, hence, it gets the least weight. Figure 10(a) compares the performance of $g_{\boldsymbol{\theta}_2}$'s on BSD68 ($\sigma = 25$) when using $(\alpha, \beta, \gamma) = (5, 1, 10)$, for the outer minimization, as proposed, and using $(\alpha, \beta, \gamma) = (1, 1, 1)$. We observe there is a significant gap between the two.

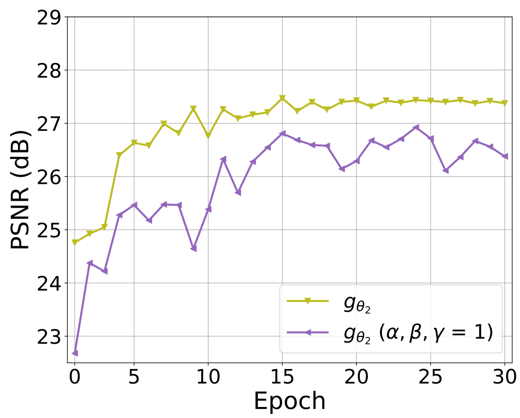


Figure 3. Ablation study on (α, β, γ)

Secondly, the output layer of $g_{\boldsymbol{\theta}_2}$ *must* have the sigmoid activation function. Note $g_{\boldsymbol{\theta}_2}$ itself can be thought of another denoiser, but since we are not training it with any target, we need to ensure the outputs of $g_{\boldsymbol{\theta}_2}$ have values between $[0, 1]$ to prevent from obvious errors of generating negative or out-of-bound pixel values. Without the sigmoid activation, it turned out all the generators cannot be trained properly at all.

Finally, using the right architectures for the generators and critics, *e.g.*, number of layers and filters, was critical since the training procedure got very sensitive to the architectural variations. Tables 5 shows the details on the architecture of our first generator, $g_{\boldsymbol{\theta}_1}$, which aims to generate noise patches. The dimension of $\boldsymbol{r}$ (the input random vector) was set to 128, and C denotes the channel of the generated noise patch. The architectures of the $g_{\boldsymbol{\theta}_2}$ and $g_{\boldsymbol{\theta}_3}$ in our generative model are equal to that of the DnCNN model (Zhang et al., 2018), however, $g_{\boldsymbol{\theta}_2}$ had 15 layers with sigmoid activation in the output layer, and $g_{\boldsymbol{\theta}_3}$ had 17 layers and linear activation in the output layer. In addition, the architectures of the two critics, $\{f_{\boldsymbol{w}_1}, f_{\boldsymbol{w}_2}\}$, in our generative model are given in Table 6.

For training, we carry out the random cropping of the given patches to the size of 64×64, and the data augmentation was done by flipping the cropped patches horizontally and vertically. For optimization, we used Adam (Kingma & Ba, 2015) optimizer for the three generators and RMSProp (Tieleman & Hinton, 2012) optimizer for the two critics. The initial learning rates were set to 0.0004 and 0.0005 for Adam and RMSProp, respectively. Also, the learning rate decay, dropping

Table 5. Architectural details on $g_{\boldsymbol{\theta}_1}$.

Input shape : $(128,)$		Details of DeConv layer				
Layer Num	Layer composition	Input channel	Output channel	Kernel size	Stride	Padding
1	DeConv + BatchNorm + ReLU	128	64	4	1	0
2	DeConv + BatchNorm + ReLU	64	32	4	2	1
3	DeConv + BatchNorm + ReLU	32	16	4	2	1
4	DeConv + BatchNorm + ReLU	16	8	4	1	1
5	Conv + Tanh	8	C	4	2	1
Output shape : (64x64xC)		-				

Table 6. Architectural details on the critics, $\{f_{\boldsymbol{w}_1}, f_{\boldsymbol{w}_2}\}$.

Input shape : (64x64xC)		Details of Conv layer					Details of LeakyReLU
Layer Num	Layer composition	Input channel	Output channel	Kernel size	Stride	Padding	α
1	Conv + BatchNorm + LeakyReLU	C	128	4	2	1	0.2
2	Conv + BatchNorm + LeakyReLU	128	256	4	2	1	
3	Conv + BatchNorm + LeakyReLU	256	512	4	2	1	
4	Conv	512	1	4	1	0	-
Output shape : (64x64x1)		-					-

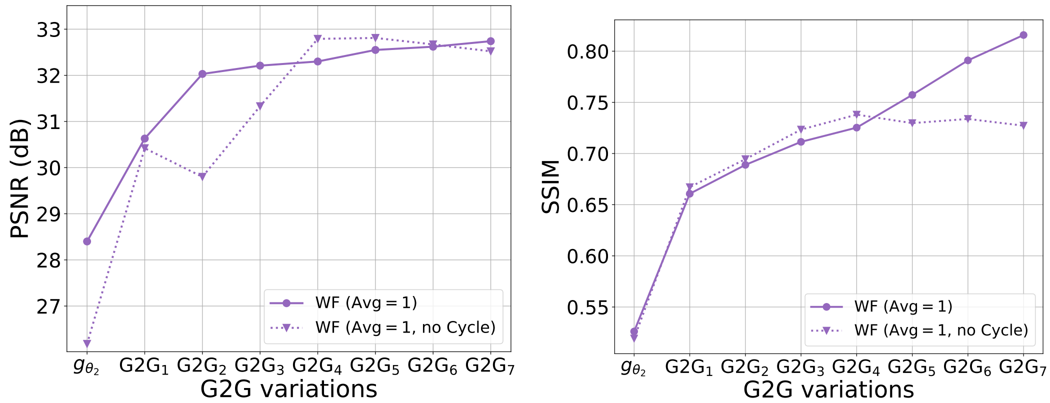
the learning rate linearly starting from epoch 10, is applied to the Adam optimizer. The parameter clipping was done for the critics and the range was set to $[-0.02, 0.02]$, and the number of training iterations for the critics was 5. The total number of training epochs was 30 and the mini-batch size was 64. The pseudo algorithm for training a generative model is in Algorithm 1

Algorithm 1 Training a generative model, all experiments in this paper used the defaults values, $n_{critic} = 5$, $n_{epoch} = 30$, m = 64, $\alpha_g = 4e^{-4}$, $\alpha_{critic} = 5e^{-5}$, $\alpha = 5$, $\beta = 1$, $\gamma = 10$

Require $\mathcal{D}$, λ
$\mathcal{N} \leftarrow$ NoisePatchExtraction($\mathcal{D}, \lambda$)
for $ep_{GAN} \leftarrow 1, n_{epoch}$ **do**
  Sample $\{n^{(i)}\}_{i=1}^m \sim \mathcal{N}$, $\{r^{(i)}\}_{i=1}^m \sim N(0, I)$, $\{Z^{(i)}\}_{i=1}^m \sim \mathcal{D}$
  for $ep_{critic} \leftarrow 1, n_{critic}$ **do**
    $g_{w_1} \leftarrow \nabla_{w_1}[\mathcal{L}_n(\theta_1, w_1)]$
    $g_{w_2} \leftarrow \nabla_{w_2}[\mathcal{L}_Z(\theta_1, \theta_2, w_2)]$
    $w_1 \leftarrow Clip(w_1 + \alpha_{critic} \cdot Adam(w_1, g_{w_1}), -c, c)$
    $w_2 \leftarrow Clip(w_2 + \alpha_{critic} \cdot Adam(w_2, g_{w_2}), -c, c)$
  end for
  $g_{\theta_1}, g_{\theta_2}, g_{\theta_3} \leftarrow \nabla_{\theta_1,\theta_2,\theta_3}[\, \alpha\mathcal{L}_n(\theta_1, w_1) + \ \beta\mathcal{L}_Z(\theta_1, \theta_2, w_2) + \gamma\mathcal{L}_{\text{cyc}}(\theta_2, \theta_3)\,]$
  $\theta_1 \leftarrow \theta_1 - \ \alpha_g \cdot Adam(\theta_1, g_{\theta_1})$
  $\theta_2 \leftarrow \theta_2 - \ \alpha_g \cdot Adam(\theta_2, g_{\theta_2})$
  $\theta_3 \leftarrow \theta_3 - \ \alpha_g \cdot Adam(\theta_3, g_{\theta_3})$
end for
return $g_{\theta_1}, g_{\theta_2}$

5.6.1. Ablation study on $\mathcal{L}_{\text{cyc}}$

As shown in the synthetic noise case of Figure 6(b) (manuscript), the iterative G2G training is powerful such that there is a negligible performance difference between the schemes with and without $g_{\boldsymbol{\theta}_2}$, when the number of iterations is sufficiently large. Consequently, the cycle loss $\mathcal{L}_{cyc}$ also does not have significant effect in the final performance for the synthetic noise case. However, for the real noise case, $\mathcal{L}_{cyc}$ becomes more critical. As shown in Figure 4(a) and 4(b), on WF(Avg$=1$) dataset, we observe that when there is no $\mathcal{L}_{cyc}$ in our generative model, the final PSNR or SSIM performances cannot reach the model *with* $\mathcal{L}_{cyc}$ even after many iterations of G2G training. Hence, this result shows the necessity of $\mathcal{L}_{cyc}$.



(a) Iterative G2G (PSNR)

(b) Iterative G2G (SSIM)

Figure 4. Figure (a) and (b) compares the PSNR and SSIM performances between starting from $g_{\boldsymbol{\theta}_2}$ and $g_{\boldsymbol{\theta}_2}$(No $\mathcal{L}_{cyc}$), respectively.

5.7. Iterative GAN2GAN training of a denoiser

We do the same random cropping and data augmentation as in the generative model training. Moreover, for every minibatch in the G2G training, we generated new synthetic noisy image pairs using our trained generators as was done in the noise augmentation of (Zhang et al., 2017). Adam optimizer with an initial learning rate 0.001 was used, and the learning rate scheduling, which halves the learning rate every 20 epochs, was applied. The total number of training epochs was 50, and the mini-batch size was 4. We also stress that we set the architecture of $\hat{\mathbf{X}}_{\boldsymbol{\theta}}(\mathbf{Z})$ identical to that of 17-layers DnCNN in (Zhang et al., 2017) to make a fair comparison. The pseudo algorithm for training a generative model is in Algorithm 2

Algorithm 2 Training G2G, all experiments in this paper used the defaults values, $n_{epoch} = 50$, $\alpha_{G2G} = 1e^{-3}$

Require $\mathcal{D}$, g_{θ_1}, g_{θ_2}, ϕ, num_iter, m
for $j \leftarrow 1$, num_iter **do**
  for $ep \leftarrow 1, n_{epoch}$ **do**
    Sample $\{r_{j,1}^{(i)}, r_{j,2}^{(i)}\}_{i=1}^{m} \sim N(0, I)$, $\{Z^{(i)}\}_{i=1}^{m} \sim \mathcal{D}$
    $\boldsymbol{\phi}_j \leftarrow \arg\min_{\boldsymbol{\phi}} \mathcal{L}_{\text{G2G}}(\boldsymbol{\phi}, \hat{\mathcal{D}}_j)$,
  end for
end for
return ϕ_{num_iter}

5.8. Noise2Void (Krull et al., 2019)

We used the publicly available source code of Noise2Void (N2V) (Krull et al., 2019) to obtain the denoising results of N2V. Most of the hyperparameters were set to the default ones, but we changed three things to make a fair comparison with our method.

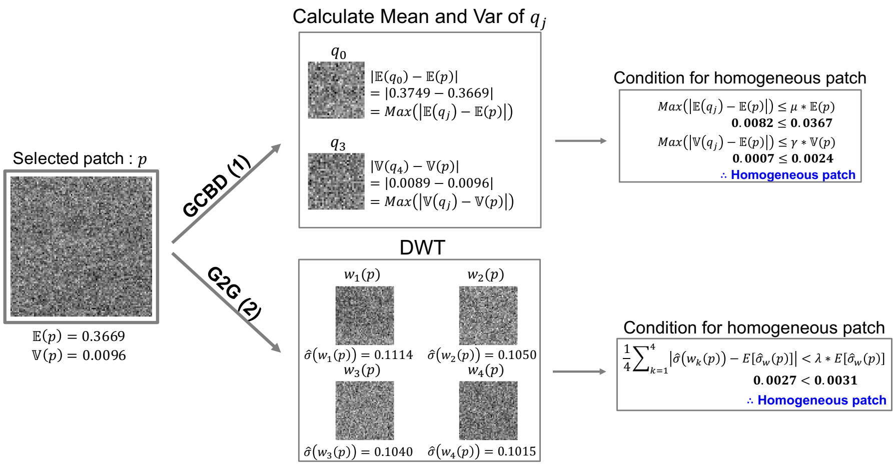
Firstly, while the CNN architecture for the original N2V was a UNet3, we used the DnCNN (Zhang et al., 2018) with 17 layers such that it has the same structure as our G2G model. Secondly, as also is done in (Krull et al., 2019), we had to use a validation set to do a proper model selection for N2V (i.e., the best epoch), while our G2G does not require any validation set (since we always use a model at the last epoch). The reason why N2V needs a validation is that its learning curve is very unstable and a proper model selection greatly affects the final denoising performance. To that end, since we used 20,500 patches with 120×120 size for training our G2G and other baselines, we divided the 20,500 patches into 18,000 training patches and 2,500 validation patches for training and selecting the best N2V model. Thirdly, we set 'mini_batch_size' to 4 (as our G2G) and 'train_steps_per_epoch' to 'num_of_training_data / mini_batch_size', hence, 4,500. Other hyperparameters are given in Table 7.

Table 7. Hyperparameters for N2V

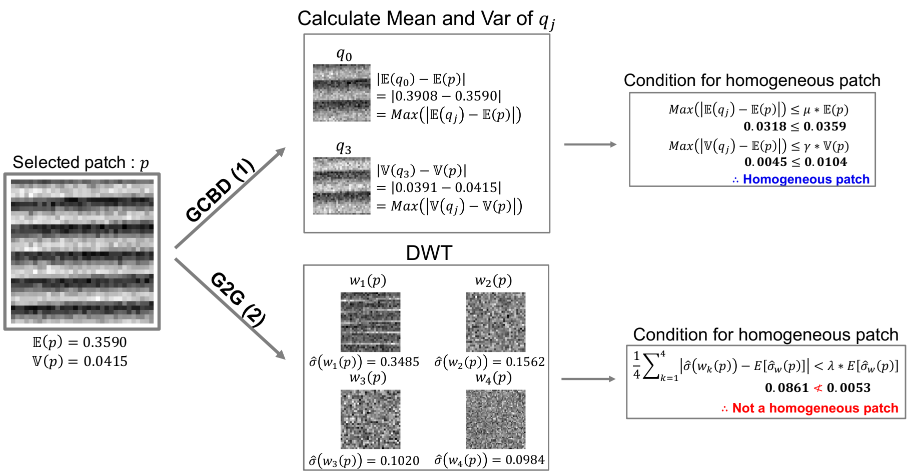
Hyperparameter	Value
train_steps_per_epoch	4,500
train_loss	'mse'
train_scheme	'Noise2Void'
train_batch_size	4
n2v_num_pix	64
n2v_patch_shape	(64,64)
n2v_manupulator	'uniform_withCP'
n2v_neighborhood_radious	'5'

6. Comparison of the patch extraction rules

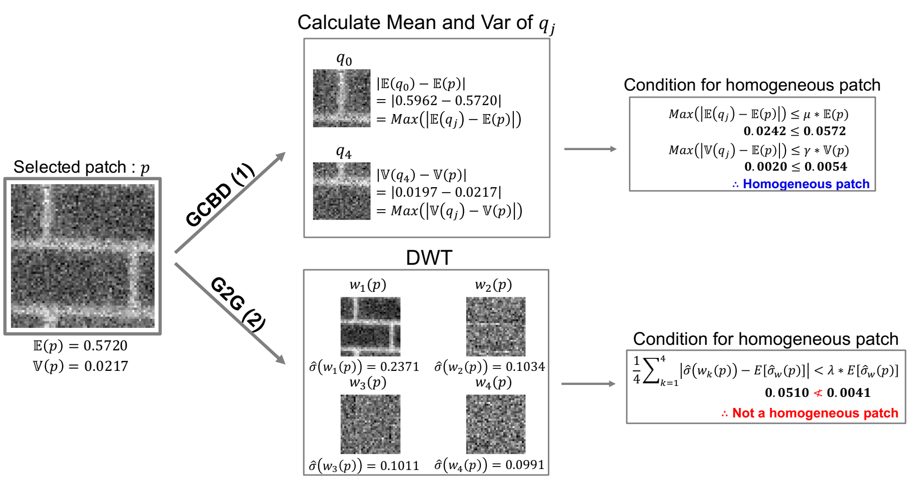
Here, we make a further, thorough comparison between the GCBD smooth patch extraction rule [Eq.(3), manuscript] and ours [Eq.(4), manuscript]. We selected three noisy patches from [Figure 2(b), manuscript] and show the decision criterion of each rule for each image Figure 5. From the figure, we can observe that while our G2G rule correctly excludes the patches in Figure 5(b) and 5(c) as non-homogeneous patches, the GCBD rule wrongly determines them also as homogeneous patches. That is, we note that since the DWT transform used in our rule can successfully disaggregate the high and low frequency components in the patches, the patches with *self-similar repeating patterns* would have significantly varying sub-band coefficient variances as shown in the figures. Hence, our rule can exclude those patches. However, in the GCBD rule, there may exist a sub-patch $\boldsymbol{q}_j$ that has similar empirical mean and variance as the original patch $\boldsymbol{p}$, thus, it may determine the patches with the self-similar repeating patterns as homogeneous as well. We believe these examples clearly show the stark difference between our rule and the GBCD rule for smooth patch extraction.



(a) Example patch determined as homogeneous by the both rules.



(b) Example patch 1 that is wrongly determined as homogeneous by the GCBD rule



(c) Example patch 2 that is wrongly determined as homogeneous by the GCBD rule

Figure 5. Noise patch extractions of GCBD (3) and G2G (4) rules.

7. Result table for real microscopy noise

We report the detailed experimental results on the real microscopy images in Table 8. We observe that Iterative G2G increases PSNR/SSIM in WF.

Table 8. Experimental results on the real microscopy dataset

Data Type	Noise Type	DnCNN-S	DnCNN$_B$	BM3D	N2V (DnCNN)	g_{θ_2}	G2G$_1$	G2G$_2$	G2G$_3$	G2G$_4$	G2G$_5$	G2G$_6$	G2G$_7$	N2C (GCBD)	N2C ((Eq.(4))
WF	Raw	35.39 /0.8738	25.43 /0.3702	26.32 /0.4012	25.31 /0.3411	28.40 /0.5261	30.63 /0.6407	32.03 /0.6889	32.21 /0.7114	32.30 /0.7253	32.56 /0.7672	32.62 /0.7910	32.74 /0.8158	31.16 /0.7493	32.26 /0.8205
	Avg = 2	36.11 /0.8969	28.36 /0.5292	29.21 /0.5642	28.23 /0.4500	29.73 /0.5844	31.84 /0.6717	32.41 /0.6920	32.80 /0.7161	32.85 /0.7500	32.90 /0.7697	32.92 /0.7783	32.85 /0.7808	31.88 /0.7575	33.23 /0.8218
	Avg = 4	37.46 /0.9182	31.32 /0.6910	32.19 /0.7202	31.28 /0.6676	31.41 /0.6580	33.32 /0.7728	33.52 /0.7974	33.71 /0.8079	33.68 /0.8091	33.85 /0.8140	33.69 /0.8088	33.79 /0.8135	34.42 /0.8665	34.79 /0.8559
	Avg = 8	39.81 /0.9374	34.63 /0.8218	35.76 /0.8444	34.85 /0.8097	34.81 /0.8084	35.16 /0.8315	35.16 /0.8315	35.27 /0.8325	35.21 /0.8321	35.27 /0.8333	35.25 /0.8330	35.22 /0.8316	36.92 /0.9126	36.86 /0.8800
	Avg = 16	42.10 /0.9569	37.82 /0.9136	39.67 /0.9293	38.75 /0.9094	36.97 /0.9086	38.97 /0.9153	38.98 /0.9174	38.84 /0.9172	38.84 /0.9170	38.87 /0.9175	38.82 /0.9181	38.82 /0.9178	38.72 /0.9110	38.92 /0.9181
Average		38.17 /0.9166	31.52 /0.6652	**32.63 /0.6919**	**31.68 /0.6365**	**32.26 /0.6971**	**33.98 /0.7664**	**34.41 /0.7855**	**34.57 /0.7970**	**34.58 /0.8067**	**34.69 /0.8203**	**34.66 /0.8258**	**34.68 /0.8324**	34.62 /0.8394	35.21 /0.8592

8. Description and the result table on Reconstructed CT dataset

The reconstructed CT dataset consists of chest and head parts of 27 pediatric extended cardiac-torso phantoms (Segars et al., 2015), which provide a highly realistic model of the human anatomy. We extracted 60 image slices from each phantom, leading to 1620 image slices in total. The dataset was generated in the following procedure. First, noiseless projection data were acquired in a parallel-beam geometry with Siddon's ray-driven algorithm (Sidky & Pan, 2008). To reduce view aliasing artifacts, the detector quarter-offset was used during a CT scan. Second, Poisson noise was generated and added to the noiseless projection data. Note that the mean number of detected photons was set to 2,500, 5,000, 7,500, and 10,000 to simulate 25%, 50%, 75%, and 100% of a normal dose, respectively. Finally, the images were reconstructed by filtered backprojection (Hsieh, 2003). To preserve fine anatomical structures in the images, the Ram-Lak filter was used as a reconstruction filter. Detailed simulation parameters are summarized in Table 9.

Table 9. Simulation parameters

Parameters	Values
Source to iso-center distance	595 mm
Source to detector distance	mm
Detector cell size	0.7 mm
Detector array size	736 x 1
Data acquisition angle	360 dares
Number of projection views	736
Reconstructed pixel width	0.67 mm
Reconstructed matrix size	512x512

We divided 27 phantoms into training and test data and the phantom number for each dataset is in Fig 10. Also, We visualized the first image of Female 1 in Fig 6. We can clearly see that each dose has a different noise level, and the noise is source independent and correlated. Finally, Table 11 shows the details of experimental results on Reconstructed CT dataset.

Table 10. Training and test data information of Reconstructed CT dataset

	Training data	Test data
Female	1,3,4,5,6,7,8,9,10,11	13,14,15
Male	1,2,3,4,5,6,7,8,9,10,11	12,13
# of images	21x60 = 1260	60x5 = 300

Table 11. Experimental results on the reconstructed CT

Data Type	Noise Type	N2C (UNnet)	DnCNN$_B$	BM3D	N2V (UNet)	g_{θ_2}	G2G$_1$	G2G$_2$	G2G$_3$
Reconstructed CT	Dose 25	48.43 /0.9609	35.50 /0.6055	42.40 /0.7575	31.57 /0.5416	34.66 /0.5759	40.49 /0.8301	46.04 /0.9579	47.47 /0.9707
	Dose 50	49.07 /0.9600	38.48 /0.7440	45.14 /0.8510	34.17 /0.6931	37.70 /0.7202	43.38 /0.9063	47.78 /0.9702	48.06 /0.9705
	Dose 75	49.45 /0.9591	40.09 /0.8111	46.55 /0.8929	35.90 /0.7756	39.49 /0.7919	44.96 /0.9320	48.80 /0.9744	49.20 /0.9733
	Dose 100	49.63 /0.9565	41.19 /0.8513	47.48 /0.9169	37.26 /0.8118	40.75 /0.9350	46.11 /0.9492	49.19 /0.9760	48.83 /0.9718
Average		49.15 /0.9591	38.82 /0.5730	**45.39 /0.8546**	**34.73 /0.7055**	**38.15 /0.7558**	**43.74 /0.9044**	**47.95 /0.9696**	**48.39 /0.9715**

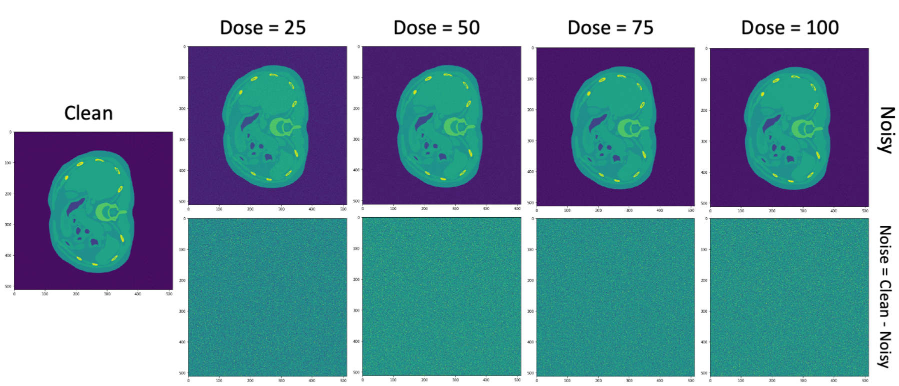


Figure 6. Clean, noisy and noise images from Reconstructed CT

9. Analysis on real microscopy image

In this section, we analyze why our G2G also works well on the real microscopy image dataset (WF) (Zhang et al., 2019). although the source-dependent noise does not satisfy our assumption on the noise. The real microscopy image dataset consists of three different types of dataset, which are Wide-Focal(WF), Two-Photon(TP) and Con-Focal(CF), and It is generally known that the real noise follows the Poisson-Gaussian model (Zhang et al., 2019),

$$Z_i = x_i + N_i, \quad i = 1, 2, \ldots, \tag{11}$$

in which $N_i \sim \mathcal{N}(0, \sigma_i^2)$ and

$$\sigma_i^2 = \alpha x_i + \sigma^2 \tag{12}$$

with a scaling factor $\alpha > 0$. Thus, the noise variance depends on the underlying clean source pixel value, and α determines the level of the dependence.

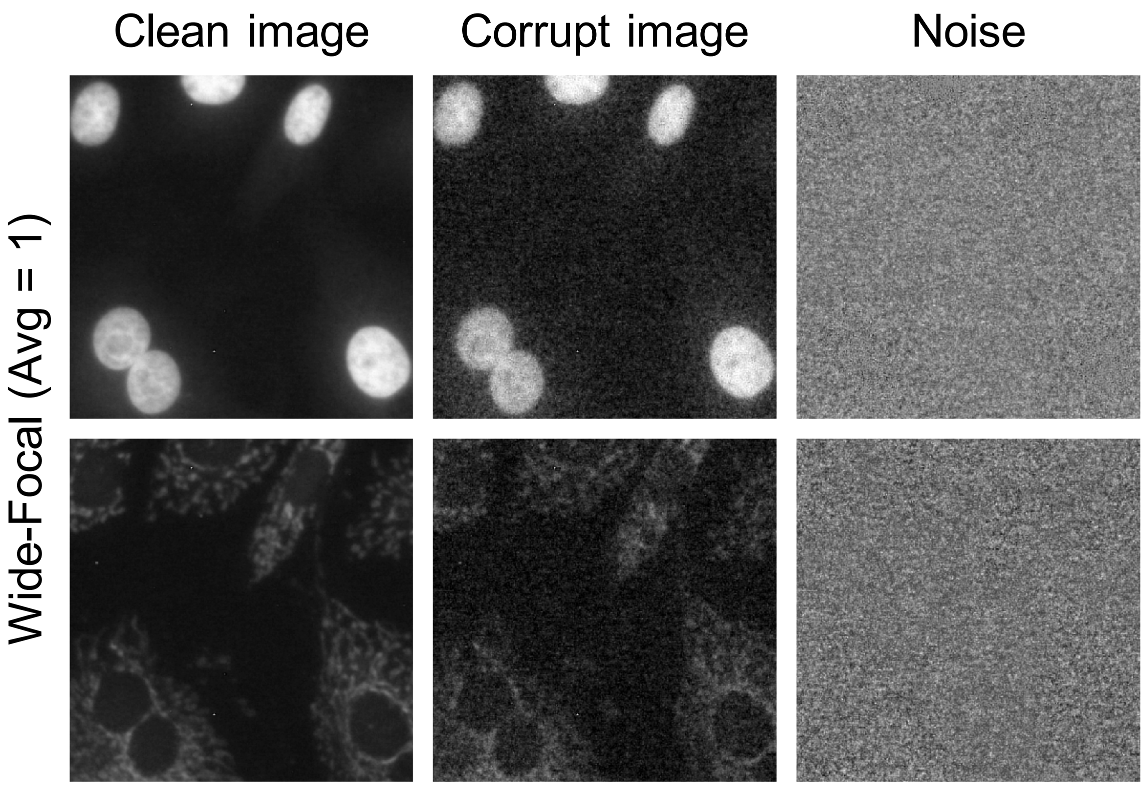


Figure 7. Clean, noisy and noise images from the WF set. (Best viewed in PDF.)

In Table 8, we observe that our G2G performs well for WF compared to other baselines, hence, we visualize clean, noisy, noise images from each set and examine if there are any notable difference in the noise distributions. Figure 7 shows two image samples (Avg= 1 cases) from the Wide-Focal (WF) set. The noise images are obtained by subtracting the clean images from its noisy versions. Note even though the intensities in the source images change significantly among pixels (particularly for the top image), the noise images do not show any source-dependent patterns. Hence, we can deduce that α may be small for the WF images. Also, we could see that there is a correlated pattern in the noise. We belive that these back the good performance of G2G for the WF set.

Figure 8, on the other hand, visualizes an image from TP set for Avg= $\{1, 16\}$ cases. Comparing with Figure 7, we can clearly see the source-dependent patterns in the noise images, particularly severely for the Avg $= 1$ case. Also, CP set showed the similar source-dependent noise patterns. This source-dependent noise is not in our assumption so we did not apply GAN2GAN to TP and CP. However, we want to stress out that the source independent real noise also exists and GAN2GAN shows the best result compared to any other baselines.

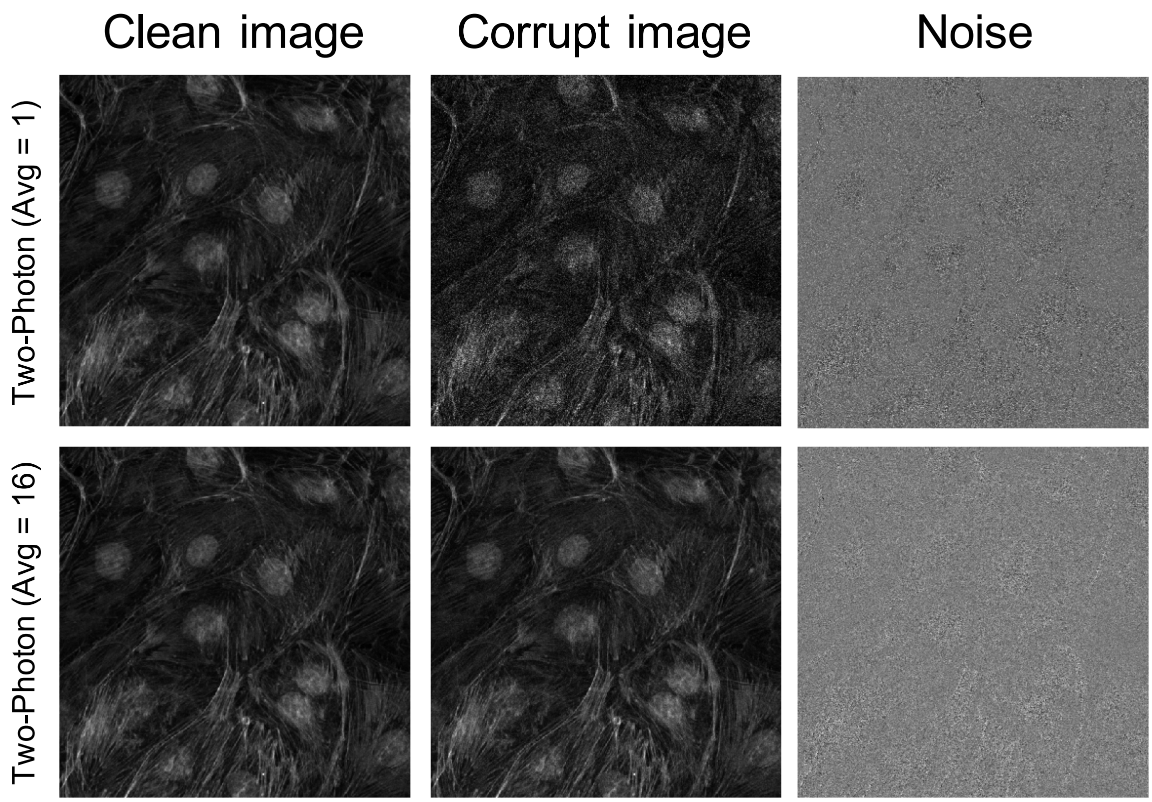


Figure 8. Clean, noisy and noise images from the TP set. (Best viewed in PDF.)

10. Ablation study

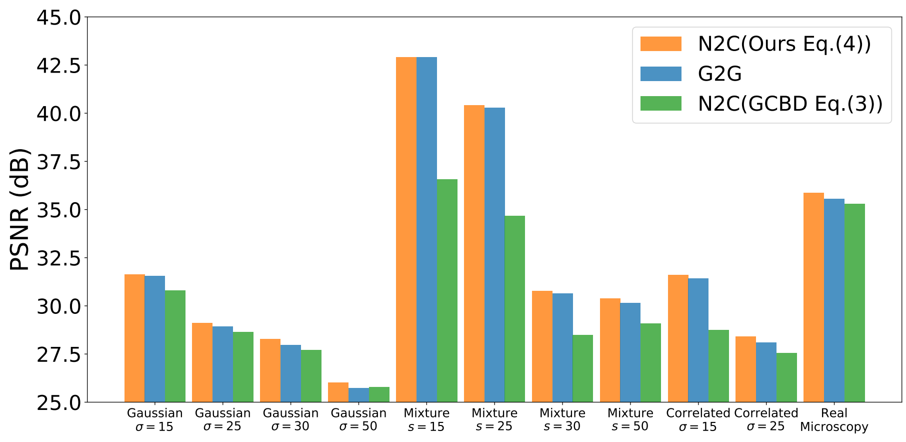
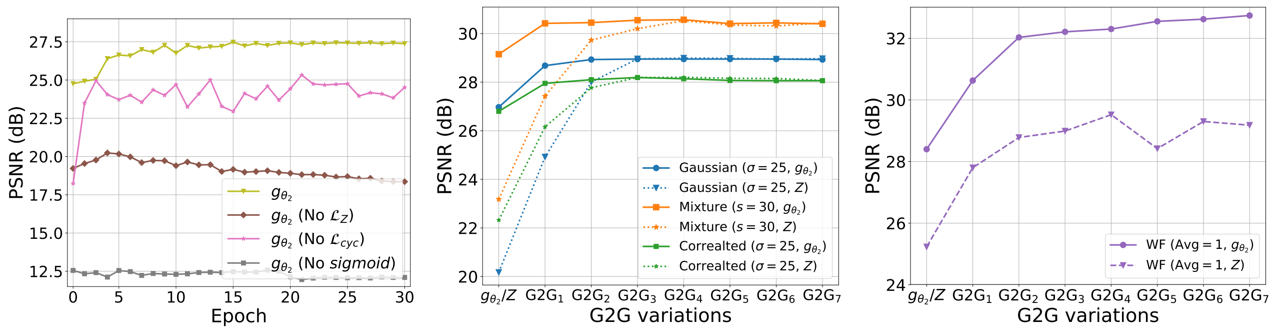


Figure 9. Effect of noise patch extraction rule.

Noise patch extraction Here, we evaluate the effect of the noisy patch extraction rules (3) and (4) (in the manuscript) in the final denoising performance. Figure 9 compares the PSNR of N2C(GCBD Eq.(3)), a re-implementation of (Chen et al., 2018), N2C(Ours Eq.(4)) and the best G2G, for each dataset.

We note neither source code nor training data of (Chen et al., 2018) is publicly available, and the PSNR in (Chen et al., 2018) could not be reproduced (with the exact same η and γ as in (Chen et al., 2018)). From the figure, we clearly observe the significant gap between N2C(Our Eq.(4)) and N2C(GCBD Eq.(3)), particularly when the noise is not white Gaussian. Moreover, our *pure* unsupervised G2G with (4) even outperforms N2C(GCBD Eq.(3)) that utilizes the clean target images.

Generative model and iterative G2G training Figure 10(a) shows the PSNRs of $g_{\boldsymbol{\theta}_2}$ on BSD68/Gaussian($\sigma = 25$) trained with three variations; "No $\mathcal{L}_{\mathbf{Z}}$" for no $f_{\mathbf{w}_2}$, "No $\mathcal{L}_{\text{cyc}}$" for no (7) in the manuscript and $g_{\boldsymbol{\theta}_3}$, and "No sigmoid" for no sigmoid activation at the output layer of $g_{\boldsymbol{\theta}_2}$. We confirm that our proposed architecture achieves the highest PSNR for $g_{\boldsymbol{\theta}_2}$, the sigmoid activation and $f_{\mathbf{w}_2}$ are essential, and the cylce loss (7) in the manuscript is also important. Achieving a decent PSNR for $g_{\boldsymbol{\theta}_2}$ is beneficial for saving the number of G2G iterations and achieving high final PSNR. More detailed analyses on the generative model architecture are in the S.M. Figure 10(b) and 10(c) show the effect of the quality of the



(a) Variations of $g_{\boldsymbol{\theta}_2}$ (b) Iterative G2G (Synthetic) (c) Iterative G2G (Real microscopy)

Figure 10. Ablation studies. (b) and (c) compare the performances between starting from $g_{\boldsymbol{\theta}_2}$ and $\mathbf{Z}$.

initial estimate for the iterative G2G training. From Figure 1, one may ask whether $g_{\boldsymbol{\theta}_2}$ is indeed necessary, since even when $\sigma_0 \approx \sigma$, the iterating the Noisy N2N can mostly achieve the upper bound. Hence, for samples of synthetic and real microscopy data, we evaluate how G2G_j performs when the iteration simply starts with $\mathbf{Z}$. Figure 10(b) shows a somewhat surprising result that for synthetic noises, starting from $\mathbf{Z}$ achieves essentially the same performance as starting from $g_{\boldsymbol{\theta}_2}$ with a couple more G2G iterations. However, for real microscopy noise case in Figure 10(c), WF(Avg$=$ 1) in which starting from $\mathbf{Z}$ achieves far lower performance than starting from $g_{\boldsymbol{\theta}_2}$, justifying our generative model for attaining the initial noisy estimate.

11. Visualizations

11.1. Visualization of $\tilde{\mathrm{Z}}$

Figure 11 and 12 visualize the simulated noisy image pairs $(\hat{\mathbf{Z}}_1, \hat{\mathbf{Z}}_2)$, generated from our generative model, for synthetic and real noise cases, respectively. A close examination shows that the images are not simple copies of the original noisy image $\mathbf{Z}$ but are successfully synthesized with the independent noise processes.

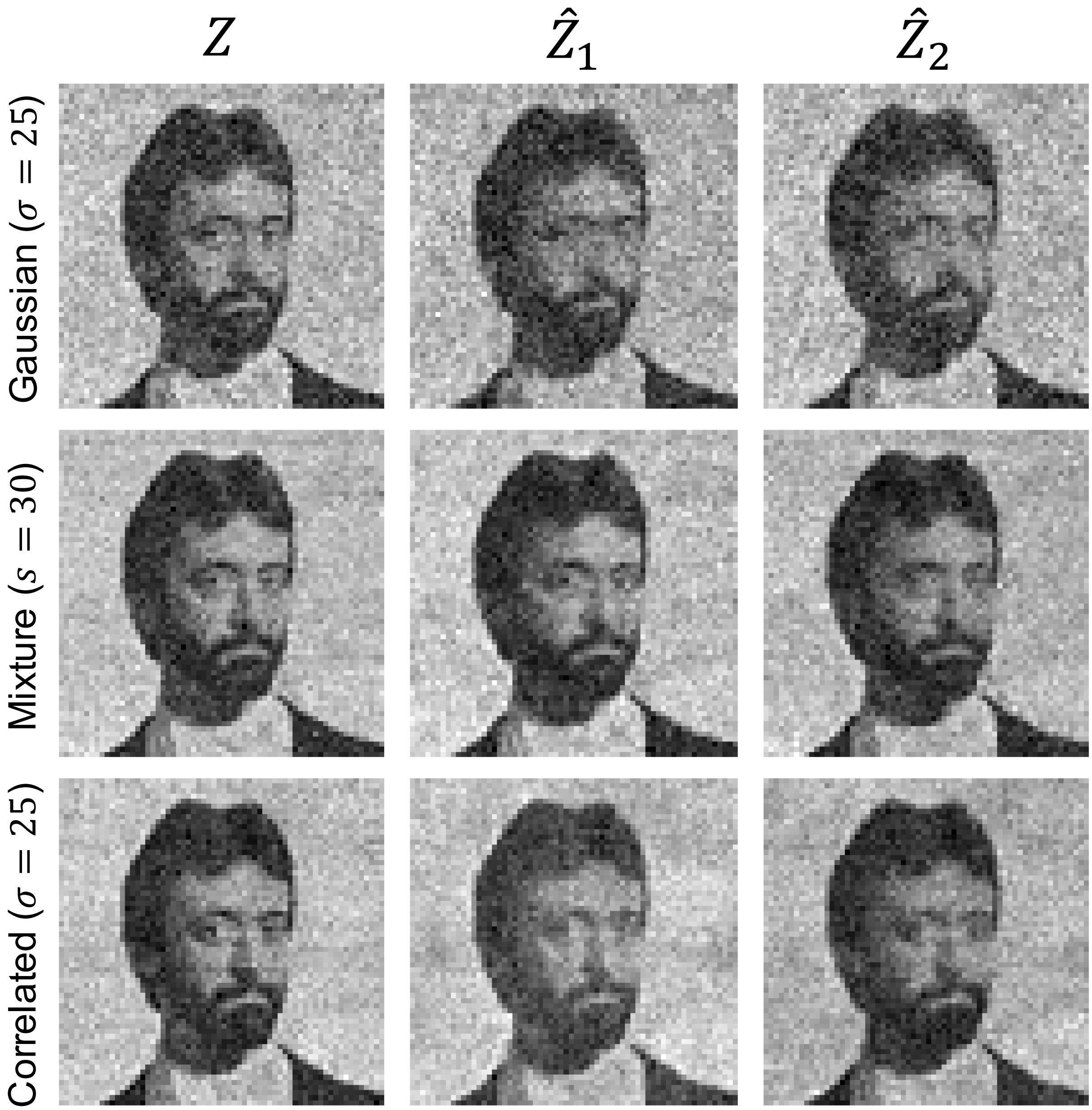


Figure 11. Visualizations of synthesized synthetic noisy image pairs.

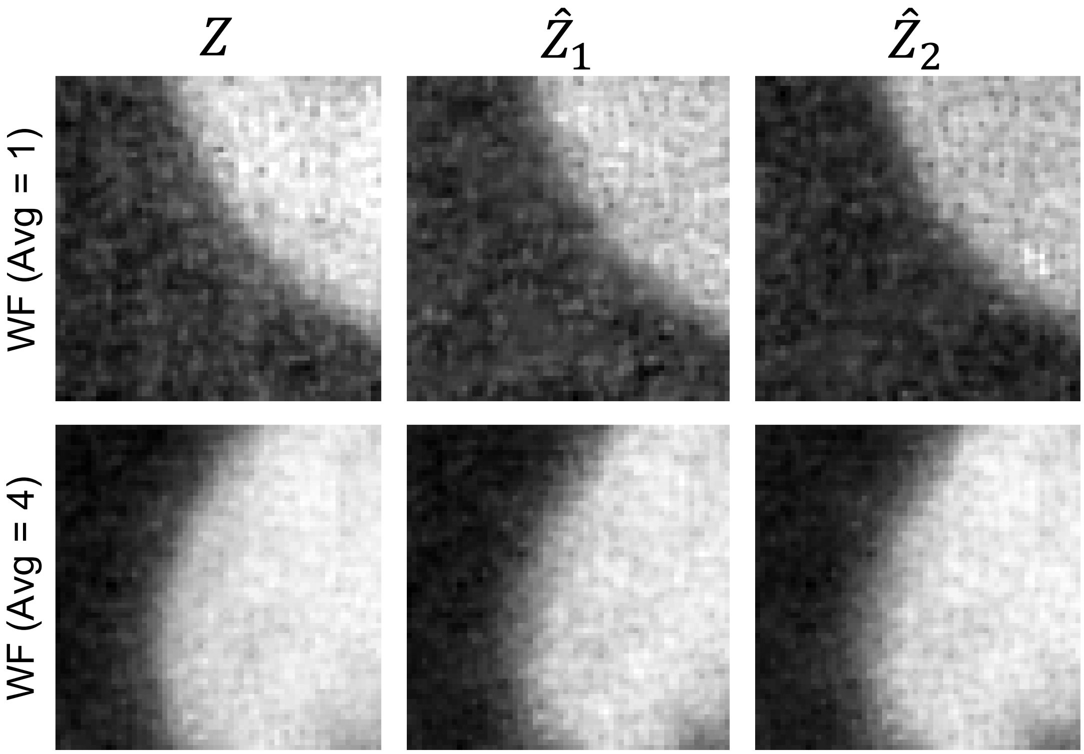


Figure 12. Visualizations of synthesized real noise image pairs.

11.2. Visualization of denoised images on BSD68

Figure 13 visualizes the denoising results of a BSD68 image for different types of noise. Note the clear difference in the noise characteristics for Gaussian, mixture, and correlated noises. The visualization of $G2G_3$ certiainly seems better than N2V and BM3D, in line with the PSNR results. DnCNN-B and Noise2Noise use more information than $G2G_3$, but the visualzation as well as the PSNR of $G2G_3$ are comparable to those of the two methods.

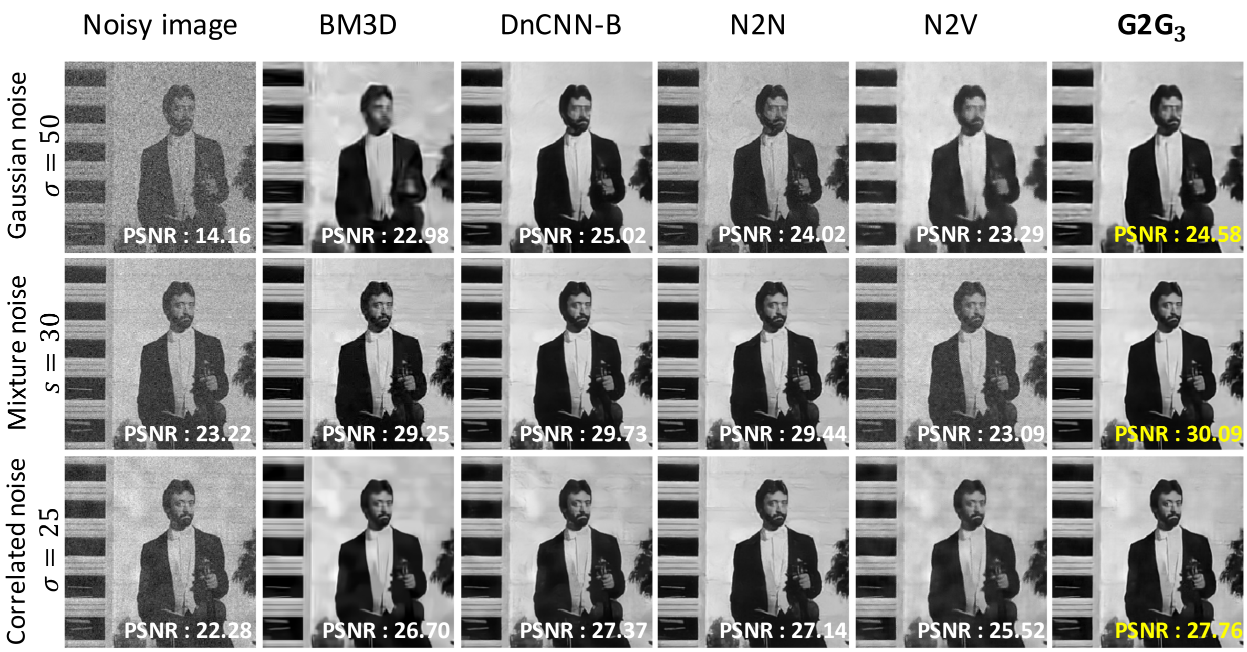


Figure 13. Denoising results on the synthetic noise images.

11.3. Additional visualizations on the real microscopy images

We also visualize additional denoised images of WF images in Figure 14. We can see that the denoising results of the baselines for WF (Avg = 1) are very noisy, but G2G$_3$ shows relatively clean denoising results than others.

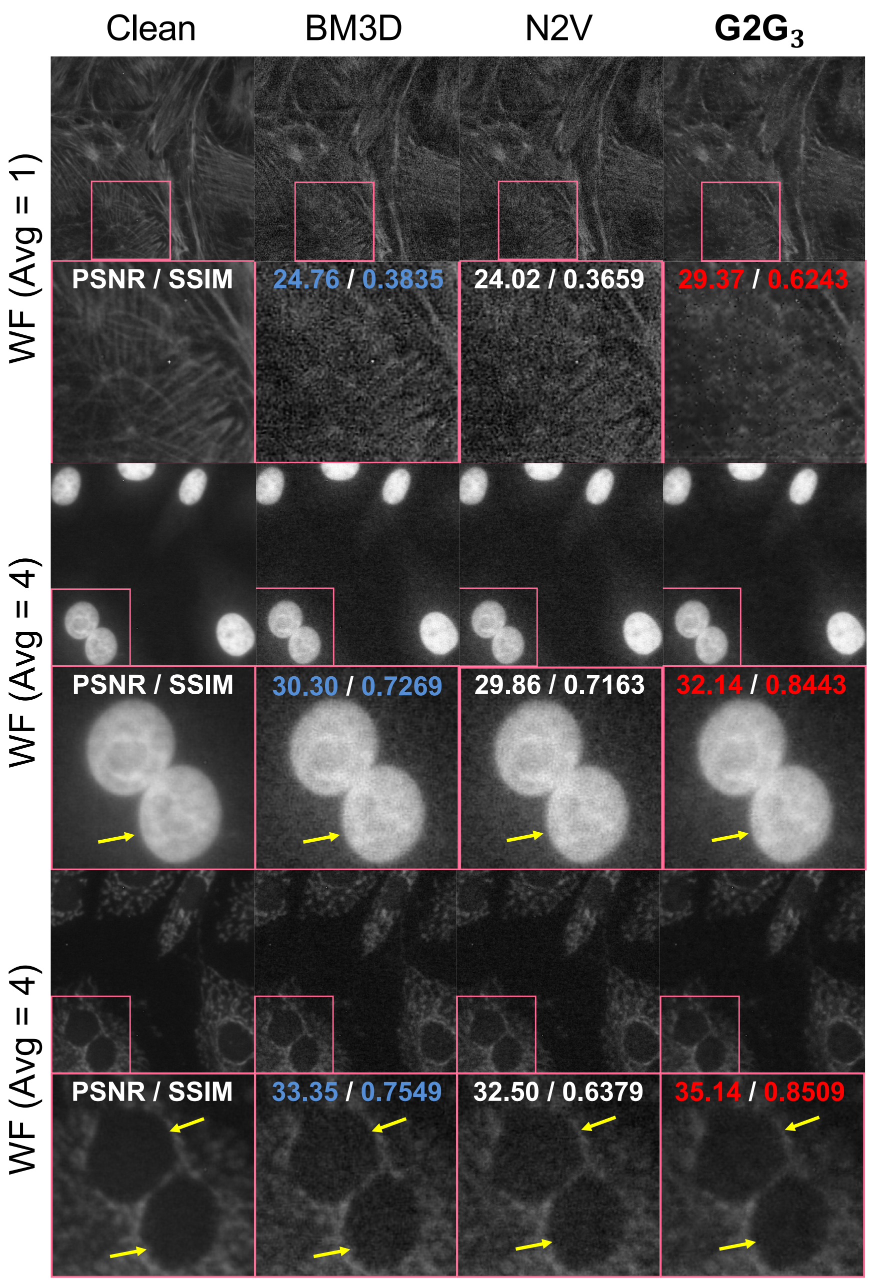


Figure 14. Denoising results on the real noisy microscopy images.

11.4. Additional visualizations on the Reconstructed CT

We visualize the denoising result of a Reconstructed CT image in Figure 15. We observed that BM3D and N2V shows a still noisy result on this image but G2G$_3$ shows a clearly denoised result.

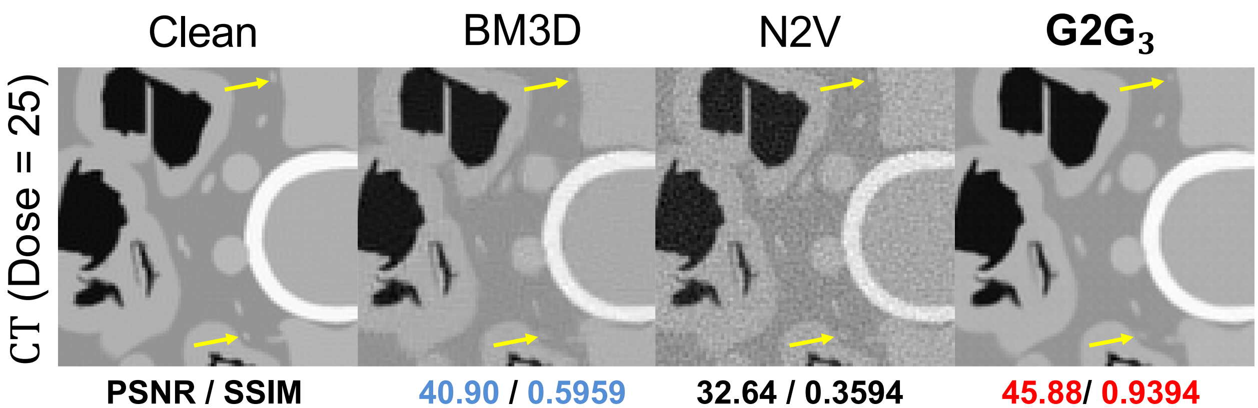


Figure 15. Denoising results on Reconstructed CT images.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017.

Batson, J. and Royer, L. Noise2self: Blind denoising by self-supervision. In *International Conference on Machine Learning (ICML)*, 2019.

Buades, A., Coll, B., and Morel, J. M. A review of image denoising algorithms, with a new one. *SIAM Journal on Multiscale Modeling and Simulation: A SIAM Interdisciplinary Journal*, 2005.

Chen, J., Chen, J., Chao, H., and Yang, M. Image blind denoising with generative adversarial network based noise modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. Image Processing*, 16(8):2080–2095, 2007.

Donoho, D. and Johnstone, I. Adapting to unknown smoothness via wavelet shrinkage. *Journal of American Statistical Association*, 90(432):1200–1224, 1995.

Elad, M. and Aharon, M. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Processing*, 54(12):3736–3745, 2006.

Hsieh, J. *Computed tomography: principles, design, artifacts, and recent advances*, volume 114. SPIE press, 2003.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Krull, A., Buchholz, T.-O., and Jug, F. Noise2void-learning denoising from single noisy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Laine, S., Karras, T., Lehtinen, J., and Aila, T. High-quality self-supervised deep image denoising. In *Advances in Neural Information Processing Systems*, pp. 6968–6978, 2019.

Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., and Aila, T. Noise2noise: Learning image restoration without clean data. In *International Conference on Machine Learning (ICML)*, 2018.

Lunz, S., Öktem, O., and Schönlieb, C.-B. Adversarial regularizers in inverse problems. In *Neural Information Processing Systems (NeurIPS)*, 2018.

Mairal, J., Bach, F., Ponce, J., Sapiro, G., and Zisserman, A. Non-local sparse models for image restoration. In *International Conference on Computer Vision (ICCV)*, 2009.

Roth, S. and Black, M. Field of experts. *International Journal of Computer Vision*, 82(2):205–229, 2009.

Segars, W., Norris, H., Sturgeon, G. M., Zhang, Y., Bond, J., Minhas, A., Tward, D. J., Ratnanather, J., Miller, M., Frush, D., et al. The development of a population of 4d pediatric xcat phantoms for imaging research and optimization. *Medical physics*, 42(8):4719–4726, 2015.

Sidky, E. Y. and Pan, X. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. *Physics in Medicine & Biology*, 53(17):4777, 2008.

Soltanayev, S. and Chun, S. Y. Training deep learning based denoisers without ground truth data. In *Advances in Neural Information Processing Systems*, pp. 3257–3267, 2018.

Tieleman and Hinton, G. RMSProp: Divide the gradient by a running average of its recent magnitude. In *Lecture Note 6-5, University of Toronto*, 2012.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Yeh, R., Lim, T. Y., Chen, C., Schwing, A. G., Hasegawa-Johnson, M., and Do, M. N. Image restoration with deep generative models. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Processing*, 26(7):3142 – 3155, 2017.

Zhang, Y., Tian, Y., Kong, Y., Zhong, B., and Fu, Y. Residual dense network for image restoration. *arXiv preprint arXiv:1812.10477*, 2018.

Zhang, Y., Zhu, Y., Nichols, E., Wang, Q., Zhang, S., Smith, C., and Howard, S. A poisson-gaussian denoising dataset with real fluorescence microscopy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Zhussip, M., Soltanayev, S., and Chun, S. Y. Extending stein's unbiased risk estimator to train deep denoisers with correlated pairs of noisy images. In *Advances in Neural Information Processing Systems*, pp. 1465–1475, 2019.

Zoran, D. and Weiss, Y. From learning models of natural image patches to whole image restoration. *International Conference on Computer Vision (ICCV)*, 2011.