Posterior Sampling for Image Restoration using Explicit Patch Priors

Roy Friedman  
Yair Weiss
School of Computer Science and Engineering
The Hebrew University of Jerusalem
Jerusalem, Israel
{roy.friedman,yair.weiss}@mail.huji.ac.il

Abstract

Almost all existing methods for image restoration are based on optimizing the mean squared error (MSE), even though it is known that the best estimate in terms of MSE may yield a highly atypical image due to the fact that there are many plausible restorations for a given noisy image. In this paper, we show how to combine explicit priors on patches of natural images in order to sample from the posterior probability of a full image given a degraded image. We prove that our algorithm generates correct samples from the distribution $p(x|y) \propto \exp(-E(x|y))$ where $E(x|y)$ is the cost function minimized in previous patch-based approaches that compute a single restoration. Unlike previous approaches that computed a single restoration using MAP or MMSE, our method makes explicit the uncertainty in the restored images and guarantees that all patches in the restored images will be typical given the patch prior. Unlike previous approaches that used implicit priors on fixed-size images, our approach can be used with images of any size. Our experimental results show that posterior sampling using patch priors yields images of high perceptual quality and high PSNR on a range of challenging image restoration problems.

1. Introduction

The task of image restoration includes many longstanding problems in computer vision such as denoising, deblurring, inpainting, demosaicing and super-resolution. All these problems can be phrased as recovering a “clean image” $x$ given a “distorted image” $y$. These problems are invariably ill-posed: for a given distorted image $y$, there are often an infinite number of clean images $x$ that could have given rise to $y$.

In the recent decade, there has been continuous improvement in performance on these tasks, when measured according to the mean-squared error (MSE) between the restored image and the ground truth image. This is true for prior based methods (e.g. [1, 2, 3, 4, 5, 6, 7, 8, 9]), in which a prior over clean images $p(x)$ is combined with a likelihood $p(y|x)$, as well as deep learning methods that are trained end-to-end (e.g. [10, 11, 12, 13]).

In recent years, however, the limitations of using the peak signal-to-noise ratio (PSNR) or related evaluation scores such as the structural similarity index measure (SSIM) have become increasingly clear. Often, methods that produce the highest PSNRs yield restored images of low perceptual quality: they are often more blurred and of lower contrast compared to the ground-truth images. As pointed out by Blau and Michaeli [15], this trade-off between high PSNR and perceptual quality is due to the in-
The inherent ill-posedness of the image restoration problem. To illustrate the trade-off, consider the toy image restoration problem shown in Figure 0.1. The images are sampled from a Gaussian distribution $p(x)$ which generates shoes of different colors. Assume that we know $p(x)$ exactly and are given a new distorted image $y$ obtained by converting from color to gray scale and adding noise. How should $x$ be restored from $y$? It is well known that the optimal algorithm in terms of mean squared error is the algorithm that returns the conditional mean of $x|y$ (17). In this example, since the distribution is Gaussian, this is equivalent to calculating the MAP solution. Figure 0.1(c) shows the MAP (or MMSE solution). While this predictor is optimal in terms of PSNR, the generated image is completely gray, lacking the color of the original images.

Rather than defining a deterministic algorithm that computes the reconstruction that is optimal in the MSE sense, we seek a method that can sample multiple plausible reconstructions given the corrupted image (cf. 18). As shown in Figure 0.1(e), if we sample from $p(x|y)$ each of the 4 plausible reconstructions of the noisy image have high perceptual quality. Thus rather than a deterministic algorithm that computes the mean reconstruction, we seek algorithms that can sample plausible reconstructions. Mathematically, this amounts to sampling from the posterior distribution $p(x|y)$ rather than finding the maximum of this distribution or calculating its mean.

The idea of using posterior samples in the context of image restoration goes back to [19] and is relatively easy to operationalize when the distribution is Gaussian (e.g. [19, 20]). Still, natural images are not Gaussian and, to the best of our knowledge, no existing algorithm can perform posterior sampling with an explicit non-Gaussian prior for a general image restoration problem. Some approaches train a deep neural network (DNN) to generate samples for a single, fixed problem [21, 22]. Other approaches use the prior implicit in a denoiser to approximately generate samples from a posterior for a fixed-size, small image [23, 24]. See Section 5 for a detailed discussion of related work.

The challenge in applying the idea of posterior sampling to image restoration is double. First, we need to define a prior probability $p(x)$ over full images that can capture the tremendous variety of natural images. Second, we need to devise an algorithm that can provably sample from $p(x|y)$ given a noisy observation $y$.

In this paper we address both of these challenges. Our fundamental insight is that while defining a prior probability over full images remains a difficult problem, significant progress has been achieved with modeling probabilities of small image patches [4, 5, 25, 26]. In fact, for some distributions such as image textures, a uniform distribution over a fixed dictionary of patches captures the distribution remarkably well [27, 28, 29, 30]. To leverage this insight, we show how to use explicit priors over patches of natural images in order to perform posterior sampling over arbitrarily sized images and arbitrary image degradations. We prove that our algorithm generates samples from $p(x|y) \propto \exp(-E(x|y))$ where $E(x|y)$ is the cost function minimized in previous patch-based approaches that compute a single restoration. Unlike previous approaches that computed a single restoration using MAP or MMSE, our method makes explicit the uncertainty in the restored images and guarantees that all patches in the restored images will be typical given the patch prior. Additionally, unlike methods based on implicit priors, our approach can be used with images of any size and gives an analytic form for $p(x)$. Our experimental results show that posterior sampling using patch priors yields images of high perceptual quality and high PSNR on a range of challenging image restoration problems.

2. Algorithm

![Figure 2.1](image)

(a) image sample 1 sample 2 difference
t=1 t=2 t=500

(b) input t=1 t=2 t=500

Denote by $x$ the unknown image, by $y = Hx + \eta$ the degraded image and by $P_i x$ the $i$th patch of the image. We assume that a patch model $p(P_i x)$ is available to us and that we can efficiently sample a patch from the posterior given a noisy observation $r$ of the same patch. We also assume that $\eta$ is Gaussian noise with known standard deviation $\sigma$ and that $H$ is given to us. Our algorithm can use any patch model, but for the sake of concreteness we first illustrate it in the case of a discrete dictionary of patches $p(P_i x = t) = \frac{1}{|D|}$ if $t \in D$ and 0 otherwise. In the discrete case, the posterior probability is also discrete and we simply choose one of the patches $d_i$ in the dictionary with a probability proportional to $p(r|d_i)$. We now ask, given
that we can sample patches from the posterior, how can we compute posterior samples over full images?

2.1. Exact Posterior Sampling with Non-Overlapping Patches

Consider a set of non-overlapping patches that are organized in a grid $g$. We can consider this set of non-overlapping patches as a single image, which allows us to define a probability distribution over images by assuming all patches are independent. We denote this distribution by $p_g(x) = \prod_{i \in g} p_{i}(x)$. Clearly sampling from this “prior” $p_g(x)$ is trivial; we only need to sample each patch independently. This is also true for posterior sampling given a noisy image that has been corrupted by independent noise at each pixel (i.e. $H$ is diagonal): we sample each patch from $p(d|x|P_gy)$ independently and in this manner obtain a sample of the whole image from $p_g(x|y)$.

Figure 2.1(a) shows posterior samples from $p_g(x|y)$ for the noisy test image shown on the left, where we use non-overlapping $8 \times 8$ patches and the dictionary is constructed from a set of 1300 training images (at each patch location, the dictionary consists of all training patches at the same location). By construction, each of the patches in the grid is realistic since it is sampled from one of the training images. Furthermore, the posterior samples capture the inherent uncertainty in the reconstruction: each sample gives a different plausible denoising of the image on the left. Despite these facts, this prior is clearly not suitable for image reconstruction. Both samples show noticeable grid artifacts.

2.2. Exact Posterior Sampling with Overlapping Patches

While the above algorithm is simple to execute, the prior $p_g(x)$ is unrealistic: in natural images, neighboring patches are not independent. To this end, we now consider the case of overlapping patches. Even when the patches are overlapping, we can divide them into multiple sets of non-overlapping patches, each of which defines a grid. For example, consider all $8 \times 8$ patches in an image with a stride of 4. These can be divided into four grids of which we will consider two: one grid contains patches whose top left corner has coordinates in $(1, 9, 17, \cdots)$ and the other contains patches whose top left corner has coordinates in $(5, 13, 21, \cdots)$. In the previous section, we saw how to perform posterior sampling for each grid separately. How can we combine these samples?

We use the well-known method of half-quadratic splitting (e.g. $[4, 3]$). We denote by $p_1(x)$ the distribution over images when considering only patches in the first grid (i.e. patches whose top left corner has coordinates in $(1, 9, 17, \cdots)$) and $p_2(x)$ the same for patches in the second grid (i.e. patches whose top left corner has coordinates in $(5, 13, 21, \cdots)$). We introduce two variables $x_1, x_2$ and define the following distribution over the two variables:

$$
\begin{align*}
p_\beta(x_1, x_2, y) &= p_1(x_1) \exp(-\frac{1}{4\sigma^2} \|Hx_1 - y\|^2) \times \\
p_2(x_2)\exp(-\frac{1}{4\sigma^2} \|Hx_2 - y\|^2) \times \\
\exp(-\beta \|x_1 - x_2\|^2) & 
\end{align*}
$$

As $\beta \to \infty$, $x_1$ and $x_2$ must equal each other. However, for a finite $\beta$ we can use the following Gibbs sampling algorithm:

- Sample $x_1$ given $x_2, y$; this corresponds to posterior sampling with nonoverlapping patches from a “noisy observation” which is a weighted average of $x_2$ and $y$.
- Sample $x_2$ given $x_1, y$; this corresponds to posterior sampling with nonoverlapping patches from a “noisy observation” which is a weighted average of $x_1$ and $y$.

Repeat

Figure 2.1(b) shows posterior samples from this algorithm for the same prior and noisy image as in Figure 2.1(a) and a fixed $\beta = 50$. In the first iterations, both $x_1$ and $x_2$ show strong grid artifacts but note that the artifacts appear in different locations in $x_1, x_2$ since they are based on different grids. After a few iterations the two images increasingly agree with each other and the grid artifacts are greatly reduced. If we continue the process, increasing $\beta$ throughout the process until $\beta \to \infty$, we will converge to a clean image where each patch in the image in both grids is sampled from the dictionary.

The algorithm can be generalized to any number of grids, $G$. To do so, we define $G$ variables $\{x_g\}$ and introduce a “splitting” term that requires them to be equal, as defined

\begin{algorithm}
\textbf{Algorithm 1 Posterior Sampling Algorithm}
\begin{algorithmic}[1]
\State \textbf{Input:} $H$, $y$, $\sigma^2$, $f(\cdot)$, $T$, $p_g(\cdot)$
\State Initialize $x_1^{(0)}$, $\ldots$, $x_G^{(0)}$
\State Initialize $t$
\For{$i = 1, 2, \ldots T$}
\For{$q = 1, 2, \ldots, G$}
\Comment{Increase $\gamma, \beta$}
\State $\beta, \gamma \leftarrow f(i)$
\State $\bar{x} \leftarrow \frac{x^{(i)}_q + x^{(i)}_{q+1}}{2}$
\Comment{Sample from a Gaussian}
\State $x^{(i)}_q \sim e^{-\beta\|x_q - \bar{x}\|^2 - \gamma \|x_q - t\|^2} e^{-\frac{1}{2\sigma^2\|y\|^2}}$
\Comment{Sample non-overlapping patches}
\State $t \sim e^{-\gamma\|x^{(i)}_{q+1} - t\|^2} p_g(t)$
\EndFor
\EndFor
\State \textbf{return:} $x_G^{(T)}$
\end{algorithmic}
\end{algorithm}
This is because the patches, which are independent in the pixelary variable $t$ and define a joint distribution over $x, t, y$ via:

$$p_n(x, t, y) \propto p_g(t) \exp\left(-\frac{1}{2\sigma^2}\|Hx-y\|^2\right) \exp(-\gamma\|x-t\|^2)$$

Now we can perform Gibbs sampling and sample $p(x|t, y)$ as if we were denoising non-overlapping patches with $t$ as the noisy image, and sample $p(t|x, y)$ by sampling from a multidimensional Gaussian.

The full algorithm is given in Algorithm 1. Note that it consists of two simple steps that are iterated: (1) sampling each patch in a grid independently and (2) sampling from a multidimensional Gaussian. When $H$ is diagonal (e.g. in the case of denoising), the algorithm can be simplified so that it only requires the non-overlapping patches sampling step at each iteration (see supplementary material for the exact algorithm). In the next section, we show that the iteration of these two simple steps corresponds to sampling from a posterior probability over full images.

3. Analysis

Our first result is for the case when the patch prior is uniform over a set of typical patches. This includes the case of a discrete dictionary over patches or the more expressive case of sparse coding dictionaries, which allows for any sparse combination of dictionary elements to also be typical (e.g. [5]).

**Theorem 1:** If the patch prior is uniform over a set of typical patches, then as $T, \beta, \gamma \to \infty$ Algorithm 1 converges to a posterior sample from $p(x|y)$ where the prior over images $p(x)$ is a uniform distribution on all images for which all patches are typical, assuming that the set of images for which all patches are typical is nonempty.

**Proof:** This follows directly from equation (2) and the fact that we are performing Gibbs sampling.

Our second result shows that for any patch prior, sample from Algorithm 1 are related to the Expected Patch Log Likelihood (EPLL) regularizer suggested by Zoran and Weiss in [4]. In [4] it was suggested to perform image restoration of full images given a patch prior $p(P_i|x)$ by minimizing the following energy function:

$$E_{EPLL}(x|y) = \lambda \|Hx - y\|^2 - EPLL(x)$$

with $EPLL(x) = \sum_i \log p(P_i|x)$ where $i$ ranges over all overlapping patches in the image. As the authors noted,

[This equation] has the familiar form of a likelihood term and a prior term, but note that $EPLL(x)$ is not the log probability of a full image. Since it sums over the log probabilities of all overlapping patches, it “double
counts” the log probability. Rather it is the expected log likelihood of a randomly chosen patch in the image.

Nonetheless, this approach has been very successful for a wide range of image restoration problems and has been used in many papers (e.g. [31, 30, 32]).

**Theorem 2:** As $T, \beta, \gamma \to \infty$ posterior samples from Algorithm 1 are samples from $p(x) \propto \exp(-f_{EPLL}(x|y))$.

**Proof:** Again, this follows from equation 2 whose right hand side approaches $e^{-f_{EPLL}(x,y)}$ as $\gamma, \beta \to \infty$.

### 4. Experiments

While our method can utilize any prior over patches, the experiments reported here use a GMM prior over image patches of size $8 \times 8 \times 3$ with 200 clusters. The GMM was trained exclusively on 5 million patches of the 200 train images in the BSD300 dataset [33]. See supplementary material for further information regarding the prior.

While our algorithm was originally devised to generate posterior samples, by replacing the sampling operations in lines 7 and 8 in Algorithm 1 with a maximization operation we can obtain an approximate MAP algorithm for $p(x|y) \propto e^{-f_{EPLL}(x)}$, i.e. an approximate minimizer for $f_{EPLL}(x|y)$. Throughout the experiments, we compare the posterior sampling version (indicated by PS-EPLL) with the MAP solutions (indicated by MAP-EPLL). As we show in the supplementary material, the MAP-EPLL algorithm gives almost identical performance to the EPLL algorithm in [4] when using the same GMM prior.

For all of the image denoising and deblurring results, we used 100 iterations of our algorithm with 32 grids with a fixed update schedule for $\beta, \gamma$. The typical runtime for both of these tasks on images from the BSD68 test set was around 15 seconds using GPU computations, and around 45 seconds using the CPU. These runtimes can be further reduced by using less iterations on less grids or by making use of parallel computations over grids. Since the denoising problem involves a diagonal $H$, we use the simpler algorithm for denoising and the full Algorithm 1 for deblurring.

To evaluate the perceptual quality of our samples, we used the well known NIQE [34] score which measures the deviation between the local statistics in an image and the ex-
Table 1. Average PSNR (higher is better) and NIQE scores (lower is better) for denoising of images with noise levels 15, 25 and 50 on the BSD68 test images.

| σ  | CBM3D | DPIR | DnCNN | MAP-EPLL | PS-EPLL | CBM3D | DPIR | DnCNN | MAP-EPLL | PS-EPLL |
|----|-------|------|-------|----------|---------|-------|------|-------|----------|---------|---------|
| 15 | 33.5±2.0 | 34.2±2.1 | 33.9±2.0 | 33.6±1.9 | 32.1±1.5 | 5.2±0.9 | 3.2±0.6 | 3.4±0.8 | 3.2±0.7 | 3.1±0.6 | 2.9±0.6 |
| 25 | 30.7±2.2 | 31.6±2.4 | 31.3±2.3 | 30.8±2.2 | 29.5±1.9 | 10.8±2.0 | 3.1±0.5 | 3.5±0.8 | 3.1±0.6 | 3.2±0.5 | 2.9±0.6 |
| 50 | 27.4±2.5 | 28.5±2.6 | 28.0±2.5 | 27.4±2.4 | 26.8±2.5 | 18.3±3   | 3.1±0.5 | 3.5±0.8 | 3±0.4    | 3.4±0.5  | 3.1±0.6 |

Table 2. Average PSNR (higher is better) and NIQE scores (lower is better) for deblurring of BSD68 test images blurred with Gaussian kernels of different scales.

| kernel          | PSNR↑ | NIQE↓ (clean 3.1±0.8) |
|-----------------|-------|-----------------------|
|                 | DPIR  | MAP-EPLL  | PS-EPLL | noisy | DPIR  | MAP-EPLL  | PS-EPLL |
| Isotropic, s = 1| 32.8±3.7 | 31.5±3.4 | 30.4±2.8 | 5.2±0.7 | 4.9±1.1 | 4.3±0.6 | 3.8±0.9 |
| Isotropic, s = 1.5 | 29.4±3.8 | 28.5±3.7 | 27.9±3.2 | 6.5±0.6 | 5.6±1.2 | 5.0±0.7 | 4.5±0.7 |
| Isotropic, s = 2 | 27.6±3.7 | 26.9±3.7 | 26.5±3.3 | 7.5±0.6 | 5.6±1.0 | 5.3±0.7 | 4.9±0.7 |
| Elliptical kernel | 29.1±3.7 | 28.2±3.6 | 27.6±2.7 | 6.8±0.5 | 6.0±1.3 | 5.4±0.7 | 4.4±0.8 |

Figure 3.2. Visual comparisons of sampling solutions for denoising from SRFlow and with our method, with σ = 25. SRFlow introduces strong artifacts and distortions, deviating quite a bit from the original images even for small noise levels, while our exact posterior sampling method does not.

In our first set of experiments, we performed image denoising on the BSD68 test set. We compare ourselves to a well known classical approach (CBM3D [35]), a state-of-the-art DNN denoiser that can handle many noise levels (DPIR [3]) and another CNN method that was trained separately for each noise level (DnCNN [36]).

As can be seen in Figure 2.2(a) and Table 1 our method for posterior sampling achieves lower NIQE (i.e. higher perceptual quality) at low noise levels, albeit at the cost of lower PSNRs. Furthermore, as can be seen in the perception-distortion plot in Figure 2.2(a), our method does not have higher RMSE and NIQE, at the same time, compared to other methods in low noise levels. In the terms described by Blau and Michaeli [15], our method is not dominated by any other (for low noise levels). Qualitatively, the generated posterior samples typically retain more textures and are sharper than other methods, as can be seen in Figure 3.1 While the posterior samples do contain more high-frequency details (such as textures), they also incur visual artifacts for higher noise levels. We attribute these artifacts to the EPLL approach, which is defined only on small patches (i.e. 8 × 8). These artifacts are commonly seen in EPLL-based methods and have been shown to vanish when using a multi-scale approach in [32], which can be incorporated in our method as well.

As there aren’t many methods that stochastically denoise (general subject) images, we also compare ourselves to SRFlow [22], originally devised to create stochastic SR solutions using normalizing flows. SRFlow can also be used to denoise images stochastically, as suggested in [22], by first down-sampling the corrupted input by a factor of 4 (using a bicubic kernel), essentially removing the noise from the image, before super resolving the image back to its original dimensions. As shown in Figure 3.2 sampling condi-
Deblurring: Our second set of experiments was non-blind image deblurring. Once again, we used the BSD68 test set, corrupted by isotropic and non-isotropic Gaussian kernels. We used 3 isotropic kernels with different scales $s = [1, 1.5, 2]$ and an elliptical Gaussian kernel with $\sigma_x = 1.5$, $\sigma_y = 1$ and a correlation of $\rho = 0.75$. After blurring the images, Gaussian noise with $\sigma = 2.5$ standard deviation was added to the images. We again compared our method to DPIR [3], as it is a plug-and-play based model. Our posterior sampling method achieves better NIQE scores for all blur kernels, as can be seen in Figure 2.2(b) and Table 2

The posterior samples are also visibly much sharper than other methods, as seen in Figure 4.1. To gain more insight into these results, we computed an approximate MMSE solution by averaging over 20 posterior samples, as can be seen in Figure 4.2. The discrepancies between the posterior samples is barely discernible as shown in Figure 4.2, but aggregation of these small changes in the high-frequencies results in the blurrier MMSE. At the same time, the MMSE solutions achieved higher PSNRs than the posterior samples while the MAP solutions achieved higher log-likelihood values, although both are blurrier than posterior samples.

HR Denoising: As previously stated, our method allows us to sample images of any size. Figure 4.3 shows that our method can be applied to high resolution images (results for 1800x2400 are shown here) without any modification (the same number of iterations and grids was used).

More examples for all of the above tasks can be found in the supplementary material, as well as results for inpainting.
5. Related Work

**Stochastic Super Resolution:** Recently, there has been a growing body of work dealing with the creation of multiple possible solutions for a given super resolution (SR) problem \([37, 18, 38]\). These often stochastic methods typically produce sharper images, with finer details, than their deterministic counterparts which can be attributed to the extreme ill-posedness of the SR problem. Furthermore, these methods are commonly generalized to other image restoration problems that truncate high frequencies, by first downsampling the degraded image and then using the SR output as the solution. However, most of these methods rely on an external mechanism in order to ensure some consistency between the high resolution output and the low resolution input and full consistency is unlikely. While these methods are stochastic they do not sample from the posterior probability, unlike our approach.

**Flow Models:** Models utilizing methods of normalizing flows \([21, 22]\) are a promising line of work and allow high flexibility while still being explicitly invertible. By training a normalizing flow model on conditional samples, as presented by \([21]\), they have been used to solve a diverse set of image restoration tasks. However, a new model must be trained anew for each problem, and the samples are not explicitly consistent with their inputs for tasks involving non-diagonal corruption matrices, \(H\). In contrast, our algorithm utilizes a single patch prior which can be used to restore any form of image corruption without any additional training and guarantees consistency by the fact that it is a posterior sample.

**Implicit Denoiser Priors:** Modern NNs trained to denoise images achieve impressive performance and are a powerful tool. Utilizing the prior implicitly defined by a single denoiser \([24, 39, 23]\) has shown promising results for various image restoration tasks on small images. Unfortunately, as the name suggests, it is not entirely clear what this implicit prior actually models, i.e. the prior \(p(x)\) used by these methods has no explicit form. Moreover, the sampling process is based on Langevin dynamics and similar methods, which can be very slow and unstable. For this reason, these approaches are almost always applied to small images (e.g. \(96 \times 96\) images for \([24]\), \(256 \times 256\) images for \([39]\)). In contrast, our algorithm is based on efficient patch sampling and can be applied to arbitrarily sized images, while still using an explicit prior.

6. Conclusion

As was pointed out by Fieguth in \([19]\) two decades ago:

[Estimates for restored images based on MAP or MMSE] are not a realistic version of the random field, and do not represent a typical or representative sample of the system being studied. Instead, what is often desired is that we find a random sample from the posterior distribution, a much more subtle and difficult problem than estimation.

The challenge in applying this idea to general, real images is the inherent difficulty of calculating and sampling from \(p(x)\) where \(x\) is a full image. In this paper, we have taken an alternative approach where we assume we have access to a good prior over local patches of natural images. This allows for posterior sampling using any prior over patches, as long as posterior sampling from each patch is possible. Using a simple GMM prior over patches, we have shown that such posterior samples are sharper and retain more of the properties typical to natural images than methods specializing in the minimization of the MSE, such as the MAP and MMSE estimates. Consistent with past observations, these improvements in the perceptual quality come at the price of the PSNR, but even after this drop the PSNR is quite high. Further still, posterior samples allow us to make explicit the uncertainty in obtaining restorations of degraded images.
Acknowledgements

We thank the Gatsby Foundation for financial support.

References

[1] Geng Ji, Michael C. Hughes, and Erik B. Sudderth. From patches to images: A nonparametric generative model. Volume 70 of Proceedings of Machine Learning Research, pages 1675–1683, International Convention Centre, Sydney, Australia, 06-11 Aug 2017. PMLR.

[2] Uwe Schmidt, Qi Gao, and Stefan Roth. A generative perspective on mrfs in low-level vision. In The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010, pages 1751–1758. IEEE Computer Society, 2010.

[3] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. CoRR, abs/2008.13751, 2020.

[4] Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In Dimitris N. Metaxas, Long Quan, Alberto Saffari, and Luc Van Gool, editors, ECCV: European Conference on Computer Vision, ECCV 2011, Barcelona, Spain, November 6-13, 2011, pages 479–486. IEEE Computer Society, 2011.

[5] Michal Aharon, Michael Elad, and Alfred M. Bruckstein. K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. Signal Process., 54(11):4311–4322, 2006.

[6] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen O. Egiazarian. Image restoration by sparse 3d transform-domain collaborative filtering. In Jaakko Astola, Karen O. Egiazarian, and Edward R. Dougherty, editors, Image Processing: Algorithms and Systems VI, San Jose, California, USA, January 28-29, 2008, volume 6812 of SPIE Proceedings, page 681207. SPIE, 2008.

[7] Tomer Michaeli and Michal Irani. Blind deblurring using internal patch recurrence. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III, volume 8691 of Lecture Notes in Computer Science, pages 783–798. Springer, 2014.

[8] Assaf Shocher, Nadav Cohen, and Michal Irani. "zero-shot" super-resolution using deep internal learning. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 3118–3126. IEEE Computer Society, 2018.

[9] Tom Tirer and Raja Giryes. Image restoration by iterative denoising and backward projections. IEEE Transactions on Image Processing, 28(3):1220–1234, 2019.

[10] Adam Kaufman and Raanan Fattal. Deblurring using analysis-synthesis networks pair. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020, pages 5810–5819. IEEE, 2020.

[11] Jakob Kruse, Carsten Rother, and Uwe Schmidt. Learning to push the limits of efficient fft-based image deconvolution. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 4596–4604, 2017.

[12] Ding Liu, Bihan Wen, Yuchen Fan, Chen Change Loy, and Thomas S. Huang. Non-local recurrent network for image restoration. CoRR, abs/1806.02919, 2018.

[13] Tobias Plötz and Stefan Roth. Neural nearest neighbors networks. CoRR, abs/1810.12575, 2018.

[14] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. In The Thirty-Seventh Asilomar Conference on Signals, Systems Computers, 2003, volume 2, pages 1398–1402 Vol.2, 2003.

[15] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 6228–6237. IEEE Computer Society, 2018.

[16] Xintao Wang, Ke Yu, Shixiang Wu, Jinxin Gu, Yihao Liu, Chao Dong, Chen Change Loy, Yu Qiao, and Xiaou Tang. Esrgan: Enhanced super-resolution generative adversarial networks, 2018.

[17] Steven M. Kay. Fundamentals of Statistical Signal Processing: Estimation Theory. Prentice-Hall, Inc., USA, 1993.

[18] Yuval Bahat and Tomer Michaeli. Explorable super resolution. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 2713–2722, Los Alamitos, CA, USA, jun 2020. IEEE Computer Society.

[19] Paul W. Fieguth. Hierarchical posterior sampling for images and random fields. In Proceedings of the 2003 International Conference on Image Processing, ICIP 2003, Barcelona, Catalonia, Spain, September 14-18, 2003, pages 821–824. IEEE, 2003.

[20] Zuzhou Cheng, Mathews Gadelha, Subhransu Maji, and Daniel Sheldon. A bayesian perspective on the deep image prior, 2019.

[21] You Lu and Bert Huang. Structured output learning with conditional generative flows, 2020.

[22] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In ECCV, 2020.

[23] Zahra Kadkhodaie and Eero P. Simoncelli. Solving linear inverse problems via deep priors and sampling. In Proceedings of the 2019 Conference on Neural Information Processing Systems, volume 32, Curran Associates, Inc., 2019.

[24] Aapo Hyvärinen, Jarmo Hurri, and Patrik O. Hoyer. Natural Image Statistics - A Probabilistic Approach to Early Computational Vision, volume 39 of Computational Imaging and Vision. Springer, 2009.
[26] Bruno Lecouat, Jean Ponce, and Julien Mairal. Fully trainable and interpretable non-local sparse models for image restoration. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, Computer Vision – ECCV 2020, pages 238–254, Cham, 2020. Springer International Publishing.

[27] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In Lynn Pocock, editor, Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001, Los Angeles, California, USA, August 12-17, 2001, pages 341–346. ACM, 2001.

[28] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics (Proc. SIGGRAPH), 28(3), August 2009.

[29] Anders Lindbjerg Dahl and Rasmus Larsen. Learning dictionaries of discriminative image patches. In BMVC, pages 1–11, 2011.

[30] Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Edge-based blur kernel estimation using patch priors. In IEEE International Conference on Computational Photography (ICCP), pages 1–8, 2013.

[31] Tobias Plotz and Stefan Roth. Benchmarking denoising algorithms with real photographs. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 2750–2759. IEEE Computer Society, 2017.

[32] Vardan Papyan and Michael Elad. Multi-scale patch-based image restoration. IEEE Transactions on Image Processing, 25(1):249–261, 2016.

[33] David R. Martin, Charless Fowlkes, D. Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In Proc. 8th Int’l Conf. Computer Vision, volume 2, pages 416–423, July 2001.

[34] Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. Making a “completely blind” image quality analyzer. IEEE Signal Processing Letters, 20(3):209–212, 2013.

[35] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE transactions on image processing : a publication of the IEEE Signal Processing Society, 16:2080–95, 09 2007.

[36] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing, 26(7):3142–3155, 2017.

[37] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models, 2020.

[38] Marcel C. Buhler, Andres Romero, and Radu Timofte. Deepsee: Deep disentangled semantic explorative extreme super-resolution. In Proceedings of the Asian Conference on Computer Vision (ACCV), November 2020.

[39] Bahjat Kawar, Gregory Vaksman, and Michael Elad. Stochastic image denoising by sampling from the posterior distribution, 2021.
Appendix A. GMM as a Patch Prior

As stated in the main text, we used a GMM prior over patches, i.e.:

\[ p(x^i) = \sum_k \pi_k \mathcal{N}(x^i | \mu_k, \Sigma_k) \]  

(5)

where \( x^i \overset{\Delta}{=} P_i x \) is the \( i \)-th patch of the image \( x \), \( K \) is the number of clusters, \( \pi_k \) are the cluster probabilities, \( \mu_k \) are the cluster means and \( \Sigma_k \) are the cluster covariance matrices.

A.1. Sampling from a GMM

As shown in Algorithm 1 from the main text, we mainly need to know how to sample from the posterior when denoising in order to sample from the posterior for a general image restoration task. To this end, let \( y^i = x^i + \eta \) where \( y^i \) is a corrupted patch, \( x^i \) is the clean patch we want to find and \( \eta \) is some zero-mean noise. For our purposes, we will assume \( \eta \sim \mathcal{N}(0, I\sigma^2) \) where \( \sigma \) is known to us, however in general the noise need not be Gaussian.

To perform denoising, we will have to find the following conditional:

\[ p(x^i | y^i) = \frac{1}{Z} e^{-\frac{1}{2\sigma^2} \| x^i - y^i \|^2} p(x) \]  

(6)

where \( Z \) is a normalization constant. The above is the product of a Gaussian with a GMM, which will be a GMM given by:

\[ p(x^i | y^i) = \sum_k p(k | y^i) \mathcal{N}
\]

(7)

\[ p(k | y^i) = \frac{\pi_k \mathcal{N}(y^i | \mu_k, \Sigma_k + I\sigma^2)}{\sum_{k'} \pi_{k'} \mathcal{N}(y^i | \mu_{k'}, \Sigma_{k'} + I\sigma^2)} \]  

(8)

\[ \mu_{x^i | y^i} = \Sigma_{x^i | y^i}^{-1} \left( \Sigma_{x^i | y^i}^{-1} \mu^0 + \frac{1}{\sigma^2} y^i \right) \]  

(9)

\[ \Sigma_{x^i | y^i} = \left( \Sigma_{x^i | y^i}^{-1} + \frac{1}{\sigma^2} I \right)^{-1} \]  

(10)

Sampling a patch from the posterior in this case can be done in two steps:

1. Sample a cluster \( k \) from \( k | y^i \), given by Equation 8

2. Sample \( x^i \) from a Gaussian with mean \( \mu_{x^i | y^i} \) and covariance \( \Sigma_{x^i | y^i} \), given by Equations 9 and 10 respectively

A.2. Quality of the Prior

We trained a GMM over 5 million patches from the train set of BSD300, using the expectation maximization (EM) algorithm, for 30 iterations. Figure A.1 shows samples drawn from the GMM (right) next to random patches from the train set (left).

Appendix B. Equivalence with EPLL

| \( \sigma \) | EPLL [4] | MAP-EPLL (ours) |
|---|---|---|
| 15 | 31.21 | 31.29 |
| 25 | 28.71 | 28.73 |
| 50 | 25.72 | 25.66 |

Table 3. Comparison of PSNR between our method and the original EPLL algorithm. As can be seen, both methods achieve nearly the same PSNR.

While the MAP version of our algorithm find a local maximum of the EPLL energy term, it is not necessarily equivalent to the EPLL algorithm. To show that the performance of both methods is similar, we used the same GMM as was used in [4] and compared the performance of both methods. The results of this experiment can be seen in Table 3 and show the performance is very similar.
Appendix C. Denoising Algorithm

Algorithm 2 Denoising Algorithm

Input: $H, y, \sigma^2, f(\cdot), T, p_y(\cdot)$
1: Initialize $x_1^{(0)}, \ldots, x_G^{(0)}$
2: Initialize $t$
3: for $i = 1, 2, \ldots, T$
4:   for $g = 1, 2, \ldots, G$
5:       $\triangleright$ Increase $\beta$
6:       $\beta \leftarrow f(i)$
7:       $\bar{x} \leftarrow \frac{x_{g-1}^{(i)}+x_g^{(i)}}{2}$
8:       $\triangleright$ Sample independent patches using Eq. 7
9:       $x_g^{(i)} \sim e^{-\beta\|x_g-\bar{x}\|^2} e^{-\frac{1}{2\sigma^2}\|x_g-y\|^2} p_y(x_g)$
10: end for
11: end for
12: return: $x_G^{(T)}$

In the main body of text we presented the general posterior sampling algorithm, however a simplified version can be used for denoising. Algorithm 2 depicts this simplified algorithm. In words, the second splitting term that was added in the original algorithm, separating the denoising algorithm. In words, the second splitting term that was be used for denoising. Algorithm 2 depicts this simplified prior sampling algorithm, however a simplified version can be used. For all of them, $\eta$ may have remained after the sampling procedure. This additional step smoothed out any grid artifacts which were of the form:

$$f_\beta(i) = \left(1 + \left(\frac{i}{\tau_\beta}\right)^{d_\beta}\right) \cdot a_\beta$$

For inpainting, a modified version of Algorithm 2 was used, where instead of spherical noise $\eta$ a diagonal covariance was assumed, i.e. $\eta \sim \mathcal{N}(0, D)$ where $D$ is a diagonal matrix. In this case, the only change is that every account of $I\sigma^2$ in Equations 8–10 needs to be exchanged with the matrix $D$, and every appearance of $I\frac{1}{\sigma^2}$ needs to be changed with $D^{-1}$.

Appendix D. Choice of Hyperparameters

For all of our experiments, the update schedules for $\beta$ and $\gamma$ were of the form:

$$f_\beta(i) = \left(1 + \left(\frac{i}{\tau_\beta}\right)^{d_\beta}\right) \cdot a_\beta$$

$$f_\gamma(i) = \left(1 + \left(\frac{i}{\tau_\gamma}\right)^{d_\gamma}\right) \cdot a_\gamma$$

Further, for some of the tasks an additional step with a much larger $\beta$ (100× the maximal value) was used - we found that this additional step smoothed out any grid artifacts which may have remained after the sampling procedure.

In each of the tasks (subtly) different schedules were used. For all of them, $d_\beta = 2.2$ was the same. The rest of the parameter settings were:

1. $\tau_\beta$: for denoising and deblurring it was 18, for inpainting 6
2. $a_\beta$: for denoising $\frac{1}{\tau_\beta}$, for inpainting 2, for deblurring 10
3. For deblurring there were the additional parameters: $\tau_\gamma = 1, d_\gamma = 0.65, a_\gamma = 0.1$

We found that the ordering of the grids in the Gibbs sampling procedure and the schedule are dependent on each other. In our experiments, the order the grids were sampled in was from the first to the last and then from the last to the first, i.e. 1, 2, $\ldots, G-1, G, G-1, \ldots, 2, 1, 2, \ldots$. For the first pass over the grids, only the previous grid in the chain was considered, as an initialization of the grids.

Finally, for the MAP solution we also returned the mean over all of the grids instead of just the last grid in the chain. We found that this typically boosted the PSNR, but also the EPLL score of the result. In all of the results displayed, the MAP had a higher EPLL score than both the MMSE and the posterior samples, as we should expect from a MAP estimate.

Appendix E. SRFlow with 8X Downsampling

In the main text, comparisons with SRFlow were given only with 4X downsampling. In this case, the downsampling may not completely remove traces of the noise before sampling again. Figure C.1 shows comparisons between SRFlow and our posterior sampling algorithm when the images are downsampled by a factor of 8 before reconstructions are sampled from SRFlow. As can be seen, SRFlow samples add significant distortions to the reconstructions while our posterior samples do not.

Appendix F. More Examples

Figures F.1–F.10 show more examples of image denoising, deblurring and inpainting using our approach.
Figure C.1. SRFlow with 8X downsampling. The left 3 rows are examples of denoisings with $\sigma = 15$ noise, the middle 3 rows contain denoisings with $\sigma = 25$ and the right 3 rows with $\sigma = 50$.

Figure F.1. More examples of image denoising with $\sigma = 15$ noise.
Figure F.2. More examples of image denoising with $\sigma = 25$ noise.

Figure F.3. More examples of image denoising with $\sigma = 50$ noise.
Figure F.4. More examples of image deblurring for a Gaussian blur kernel with scale $s = 1$

Figure F.5. More examples of image deblurring for a Gaussian blur kernel with scale $s = 1.5$
Figure F.6. More examples of image deblurring for a Gaussian blur kernel with scale $s = 2$

Figure F.7. More examples of image deblurring for an elliptical Gaussian blur kernel
Figure F.8. Example of inpainting an image with 95% of its pixels missing.

Figure F.9. Another example of denoising a high resolution with $\sigma = 25$ noise.

Figure F.10. Another example of denoising a high resolution with $\sigma = 50$ noise.