J. Vis. Commun. Image R. 41 (2016) 96–108

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Poisson inverse problems by the Plug-and-Play scheme ☆,☆☆,☆☆☆

Arie Rond *, Raja Giryes, Michael Elad

Department of Computer Science, Technion—Israel Institute of Technology, Technion City, Haifa 32000, Israel

ARTICLE INFO

Article history:
Received 6 July 2016
Revised 2 September 2016
Accepted 17 September 2016
Available online 17 September 2016

Keywords:
Poisson denoising
Poisson deblurring
Image processing

ABSTRACT

The easy-to-compute Anscombe transform offers a conversion of a Poisson random variable into a variance stabilized Gaussian one, thus becoming handy in various Poisson-noisy inverse problems. Solution to such problems can be done by applying this transform, then invoking a high-performance Gaussian-noise-oriented restoration algorithm, and finally using an inverse transform. This process works well for high-SNR images, but when the noise level is high, it loses much of its effectiveness. This work suggests a novel method for coupling Gaussian denoising algorithms to Poisson noisy inverse problems. This approach is based on a general approach termed "Plug-and-Play-Prior". Deploying this to Poisson inverse-problems leads to an iterative scheme that repeats an easy treatable convex programming task, followed by a powerful Gaussian denoising This method, like the Anscombe transform, enables to plug Gaussian denoising algorithms for the Poisson-oriented problem, and yet, it is effective for all SNR ranges.
© 2016 Elsevier Inc. All rights reserved.

1. Introduction

In an inverse problem we are given a degraded image, y, and aim to recover from it a clean image, x. The mathematical relation between the two images is given by $y = \mathcal{N}(Hx)$, where H is some linear operator and $\mathcal{N}$ is a noise operator. A popular way to handle this reconstruction is to use a Bayesian probabilistic model that contains two ingredients: (i) the measurement forward model, mathematically given by the conditional probability $P(y|x)$; and (ii) a prior distribution model for clean images, given by $P(x)$.

This work concentrates on the case of Poisson Inverse Problems (PIP), Where $\mathcal{N}$ stands for Poisson contamination. In a Poisson model for an image the gray levels of the pixels are viewed as Poisson distributed random variables. More specifically, given a clean image pixel $x[i]$, the probability of getting a noisy value $y[i]$ is given by

$$P(y[i]|x[i]) = \begin{cases} \frac{(x[i])^{y[i]}}{y[i]!}e^{-x[i]} & \text{if } x[i] > 0 \\ \delta(y[i]) & \text{if } x[i] = 0 \end{cases}, \quad (1)$$

where $\delta(\cdot)$ is the Kronecker delta function.

A known property of this distribution is that $x[i]$ is both the mean and the variance of $y[i]$. This model is relevant in various tasks such as very low light imaging, CT reconstruction [21], fluorescence microscopy [2], astrophysics [25] and spectral imaging [16]. Common to all these tasks is the weak measured signal intensity.

An important note about Poisson noise is that the SNR of the measurements is proportional to the original image intensity, given by $\sqrt{x[i]}$. Therefore the peak value of an image is an important characteristic, needed when evaluating the level of noise in the image. For high peak levels, there exist several very effective ways to solve Poisson inverse problems. Many of these methods rely on the fact that it is possible to perform an approximate transform (known as Variance Stabilized Transform - VST) of the Poisson distribution into an approximately unit variance Gaussian one, which is independent from the mean of the transformed distribution [1,12]. Since there are highly effective algorithms for Gaussian noise restoration (e.g. [5,10,18,29,7]), such methods can be used, followed by an inversion of the VST operation after the Gaussian solver [19,30].

When dealing with lower peaks, such transformations become less efficient, and alternative methods are required, which treat the Poisson noise directly (e.g. [13,23]). In recent years this direct approach has drawn considerable attention, and it seems to be very successful. This work aims at studying yet another method for Poisson inverse problem restoration that belongs to the direct approach family. The appeal of the proposed method is the fact that it offers an elegant bridge between the two families of

☆ This paper has been recommended for acceptance by Zicheng Liu.

☆☆ This research was supported by the European Research Council under EUs 7th Framework Program, ERC grant agreement 320649, by the Intel Collaborative Research Institute for Computational Intelligence, and by the Google Faculty Research Award.

☆☆☆ This research was supported by Israel Science Foundation (ISF) grant number 1770/14.

* Corresponding author.
E-mail addresses: sarikr@campus.technion.ac.il (A. Rond), raja@tauex.tau.ac.il (R. Giryes), elad@cs.technion.ac.il (M. Elad).

http://dx.doi.org/10.1016/j.jvcir.2016.09.009
1047-3203/© 2016 Elsevier Inc. All rights reserved.

methods, as it is also relying on Gaussian noise removal, applied iteratively.

This paper is organized in the following way: Section 2 introduces the plug and play approach, as presented in [28]. This scheme is also extended to be able to work with several priors in parallel. Section 3 presents our algorithm, as derived from the plug and play approach. This section explains how to integrate a custom Gaussian denoising algorithm of choice, and discusses several improvements that were added to the algorithm. Section 4 presents experiments results, and Section 5 concludes this paper by suggesting further improvements.

2. The Plug-and-Play Prior (P&PP) approach

2.1. Standard Plug-and-Play Prior

The Plug-and-Play Prior (P&PP) framework, proposed by Venkatakrishnan et al. [28], allows simple integration between inversion problems and priors, by applying a Gaussian denoising algorithm, which corresponds to the used prior. One of the prime benefits in the P&PP scheme is the fact that the prior to be used does not have to be explicitly formulated as a penalty expression. Instead, the idea is to split the prior from the inverse problem, a task that is done elegantly by the alternating direction method of multipliers (ADMM) optimization method [3], and then the prior is deployed indirectly by activating a Gaussian denoising algorithm of choice.

The goal of the P&PP framework is to maximize the posterior probability in an attempt to implement the MAP estimator. Mathematically, this translate to the following:

$$\max_{x\in R^{m\times n}} P(x|y) = \max_{x\in R^{m\times n}} \frac{P(y|x)P(x)}{P(y)} = \max_{x\in R^{m\times n}} P(y|x)P(x). \tag{2}$$

The above suggests to maximize the posterior probability $P(x|y)$ with respect to the ideal image x, which is of size $n\times m$ pixels. Taking element wise $-\ln(\cdot)$ of this expression gives an equivalent problem of the form

$$\min_{x\in R^{m\times n}} -\ln(P(x|y)) = \min_{x\in R^{m\times n}} -\ln(P(y|x)) - \ln(P(x)). \tag{3}$$

In order to be consistent with [28] we denote $l(x) = -\ln(P(y|x))$ and $s(x) = -\ln(P(x))$. Thus our task is to find x that solves the problem

$$\hat{x} = \arg\min_{x\in R^{m\times n}} l(x) + \beta s(x). \tag{4}$$

Note that y is constant in this minimization. Also, a parameter β was added to achieve more flexibility. By adding a variable splitting technique to the optimization problem we get

$$\begin{aligned} \hat{x} = &\arg\min_{x,v\in R^{m\times n}} l(x) + \beta s(v). \\ &s.t.\ x = v \end{aligned} \tag{5}$$

This problem can be solved using ADMM [3] by constructing an augmented Lagrangian which is given by

$$L_\lambda = l(x) + \beta s(v) + \frac{\lambda}{2}\|x - v + u\|_2^2 - \frac{\lambda}{2}\|u\|_2^2. \tag{6}$$

ADMM theory [3] states that minimizing (5) is equivalent to iterating until convergence over the following three steps:

$$\begin{aligned} x^{k+1} &= \arg\min_x L_\lambda(x, v^k, u^k), \\ v^{k+1} &= \arg\min_v L_\lambda(x^{k+1}, v, u^k), \\ u^{k+1} &= u^k + (x^{k+1} - v^{k+1}). \end{aligned} \tag{7}$$

By plugging L_λ we get

$$\begin{aligned} x^{k+1} &= \arg\min_x l(y|x) + \frac{\lambda}{2}\|x - (v^k - u^k)\|_2^2, \\ v^{k+1} &= \arg\min_v \frac{\lambda}{2}\|x^{k+1} + u^k - v\|_2^2 + \beta s(v), \\ u^{k+1} &= u^k + (x^{k+1} - v^{k+1}). \end{aligned} \tag{8}$$

There exist ADMM variations that modify λ at each iteration. Thus λ could be dependent on k. The second step means applying a Gaussian denoising algorithm which assumes a prior $s(v)$ on the image $x^{k+1} + u^k$ with variance of $\sigma^2 = \frac{\beta}{\lambda}$. Therefore, as already mentioned above, the formulation of the prior does not have to be known explicitly, as the corresponding Gaussian denoising algorithm can simply be used.

The first step is dependent on the targeted inversion problem. Next section shows how Poisson inverse problems are connected to this step. In this case step 1 is convex and becomes easy to compute. When handling the Poisson Denoising problem, this steps becomes even simpler because it is also separable, thus leading to a scalar formula that resembles the Anscombe transform.

The P&PP framework's convergence is analyzed in [26]. It is shown that there exist Gaussian denoisers which are guaranteed to converge. However, other denoisers may lead to good results as well, despite the lack of solid theoretical foundations.

2.2. Extension to multiple priors

We now show a simple extension of the Plug-and-Play Prior method that enables to use several Gaussian denoisers. We start from the following ADMM formulation, that follows Eq. (5)

$$\begin{aligned} &\arg\min\ l(x) + \beta_1 s_1(v_1) + \beta_2 s_2(v_2), \\ &s.t.\ x = v_1, \quad x = v_2 \end{aligned} \tag{9}$$

where s_1 and s_2 are two priors that are aimed to be used, and v_1 and v_2 are two auxiliary variables that will help in simplifying the solution of this problem. The full derivation of L_λ appears in Appendix B. Following the steps taken above in the derivation of the P&PP, we get

Step 1:

$$\begin{aligned} x^{k+1} = \arg\min_x\ & l(x) + \lambda\|x - v_1^k + u_1^k\|_2^2 \\ &+ \lambda\|x - v_2^k + u_2^k\|_2^2. \end{aligned} \tag{10}$$

As in the original Plug-and-Play-Prior scheme, this expression too is convex if $l(x)$ is convex, and also separable if $l(x)$ is separable.

Step 2:

$$\begin{aligned} v_1^k &= \arg\min_v\ \beta_1 s_1(v) + \lambda\|x^{k+1} + u_1^k - v\|_2^2 \\ v_2^k &= \arg\min_v\ \beta_2 s_2(v) + \lambda\|x^{k+1} + u_2^k - v\|_2^2 \end{aligned} \tag{11}$$

which are two Gaussian denoising steps, each using a different prior.

Step 3:

$$\begin{aligned} u_1^{k+1} &= u_1^k + x^{k+1} - v_1^{k+1} \\ u_2^{k+1} &= u_2^k + x^{k+1} - v_2^{k+1} \end{aligned} \tag{12}$$

Obviously, this scheme can be generalized to use as many priors as needed. The core idea behind this generalization is that it is not uncommon to encounter different priors that address different features of the unknown image, such as self-similarity, local smoothness or other structure, scale-invariance, and more. By merging

two such priors into the P&PP scheme, an overall benefit may be attained, as they complement each other. Another possible usage of this extension is to force the solution to reside in some convex set. This can be done by using an indicator function for $s_2(\cdot)$.

2.3. Related ideas

In the context of Gaussian deblurring, a similar derivation is adapted in [6]. Here too, the authors use a splitting of the fidelity term and a prior term, but unlike the P&PP, an explicit formulation of the prior is required. This splitting of the two terms appears also in Poisson restoration algorithms [9,11], and is shown to be a beneficial technique in this scenario as well. The work reported in [9] relies on proximal splitting, where the assumed prior is based on positivity of the unknown, along with sparse representation of the image in a dictionary of waveforms such as wavelets or curvelets. The authors use analysis and synthesis based sparsity. The work in [11] uses ADMM to enable the splitting. The authors test the Total Variation (TV), analysis and synthesis priors. In [17] a dictionary learning technique is also added, and a variable splitting technique produces the algorithm, which treats every term separately. In contrast, our work can apply any prior that is used in the context of Gaussian denoising (even an implicit one), which gives us the freedom to use leading denoising techniques and thus get to superior results.

The idea of using several prior terms has appeared previously in the context of Poisson noise in [20]. Here, the authors use a compound of TV and a wavelet domain regularization, and show improvement over using each regularization term separately. In this work any mixture of available priors that are beneficial in Gaussian denoising may be used with complete freedom and flexibility.

3. P^{4}IP algorithm

We now turn to introduce the "Plug-and-Play Prior for Poisson Inverse Problem" algorithm, P^{4}IP in short, and how it uses the P&PP framework. We also introduce the Multiple Plug-and-Play Priors for Poisson Inverse Problem (M-P^{4}IP) algorithm, which is based on using the multiple prior P&PP, as presented above. We start by invoking the proper log-likelihood function $l(x)$ into the above-described formulation, this way enabling the integration of Gaussian denoising algorithms to the Poisson inverse problems. Then two applications of our algorithm are discussed – the denoising and the deblurring scenarios.

3.1. The proposed algorithm

We denote an original (clean) image, with dimensions $m \times n$, by an $m \times n$ column-stacked vector x. Similarly, we denote a noisy image by y. The i-th pixel in x (and respectively y) is given by $x[i]$ (respectively $y[i]$). We also denote by H the linear degradation operator that is applied on the image, which could be a blur operator, down-scaling or even a tomographic projection. In order to proceed an expression for $l(x)$ should be found. As mentioned before, this is given by taking $-\ln(\cdot)$ of $P(y|x)$. When taking H into account we get

$$P(y|x) = \prod_i \frac{(Hx)[i]^{y[i]}}{\Gamma(y[i]+1)} e^{-(Hx)[i]}. \tag{13}$$

Thus, $l(x)$ is given by

$$\begin{aligned} l(x) &= -\ln(P(y|x)) = -\sum_i \ln\left(\frac{(Hx)[i]^{y[i]}}{\Gamma(y[i]+1)} e^{-(Hx)[i]}\right) \\ &= -y^T \ln(Hx) + 1^T Hx + constant. \end{aligned} \tag{14}$$

Relying on Eq. (8), the first ADMM step in matrix form is therefore

$$\arg\min_x L_\lambda = \arg\min_x -y^T \ln(Hx) + 1^T Hx + \frac{\lambda}{2}\|x - v + u\|_2^2. \tag{15}$$

This expression is convex and can be solved quite efficiently by modern optimization methods. The final algorithm is shown in Algorithm 1. λ can be changed in many forms in each iteration. The update rule $\lambda^k = \lambda^0 \cdot (\lambda_{step})^k$, where λ^0 and λ_{step} are some constants was chosen for the P^{4}IP algorithm.

Algorithm 1. P^{4}IP

Input: Distorted image y, Gaussian_denoise$(\cdot)$ function
Initialization: set $k = 0$, $u^0 = 0$, $v^0 =$ some initialization, $\lambda^0 = const$;
while !stopping criteria **do**
$x^{k+1} = \arg\min_x -y^T \ln(Hx) + 1^T Hx + \frac{\lambda^k}{2}\|x - v^k + u^k\|_2^2$
$v^{k+1} =$ Gaussian_denoise $(x^{k+1} + u^k)$ with $\sigma^2 = \frac{\beta}{\lambda^k}$
$u^{k+1} = u^k + (x^{k+1} - v^{k+1})$
$\lambda^{k+1} = \lambda^k \cdot \lambda_{step}$
$k = k + 1$
end while
Output: Reconstructed image x^k

Obviously, the Plug-and-Play Prior extension that employs several denoising methods can be used, as shown in the previous section. Such a change requires only slight modifications to Algorithm 1. Here, the update of v is done by two denoising algorithms, and the associated variance parameter is determined by the prior weight coefficients.

3.1.1. Poisson denoising

For the special case of Poisson denoising $H = I$. In this case the first ADMM step is separable, which means that it could be solved for each pixel individually. Moreover, this step can be solved by the closed form solution

$$x^{k+1}[i] = \frac{\left(\lambda\left(v^k[i] - u^k[i]\right) - 1\right) + \sqrt{\left(\lambda(v^k[i] - u^k[i]) - 1\right)^2 + 4\lambda y[i]}}{2\lambda}, \tag{16}$$

where $x^k[i]$ is the i-th pixel of x^k (and $v^k[i]$, $u^k[i]$ and $y[i]$ are the i-th pixels of v^k, u^k and y respectively). The full derivation of this step is shown in Appendix A.

A closer look at Eq. (16) reveals some resemblance to the Anscombe transform. Indeed, for the initial condition $u^0 = 0$, $v^0 = 4\left(\sqrt{\frac{3}{8}} + 1\right)$, and $\lambda = 0.25$, the transformed random variable y after (16) is applied onto it is given by $2\sqrt{\frac{3}{8} + y} + 2\sqrt{\frac{3}{8}}$. Thus the variance is the same as the one achieved by Anscombe's transform, because the two differ only by a constant. Fig. 1 shows the Anscombe transform and the one obtained by Eq. (16) with the parameters $\lambda = 0.25$, $v^k[i] - u^k[i] = 4\left(\sqrt{\frac{3}{8}} + 1\right) + c$ for $c = \{0, 3, 6, 9\}$. While the curve (16) may look like the Anscombe one, P^{4}IP is substantially different than the Anscombe transform based denoising in two ways - (i) The curve by Eq. (16) changes (locally) from one iteration to another due to the change in u and v, and (ii) P^{4}IP has no inverse transform step after the Gaussian denoising. This leads to improved restoration compared to the Anscombe transform based denoising scheme.

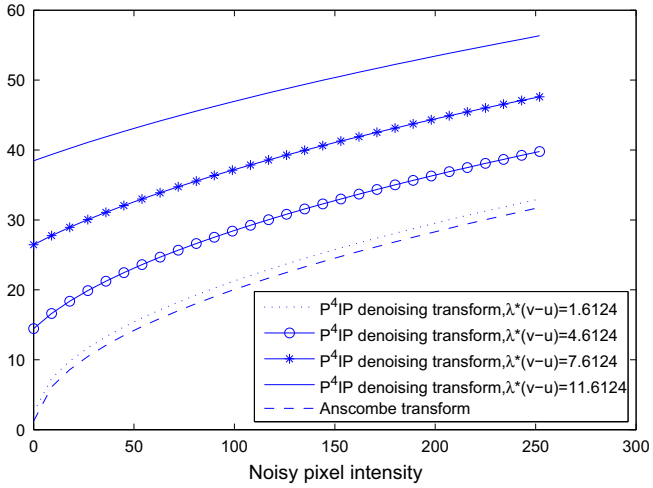


Fig. 1. P^{4}IP and Anscombe transformations as a function of input noisy pixel. $\lambda = 0.25$.

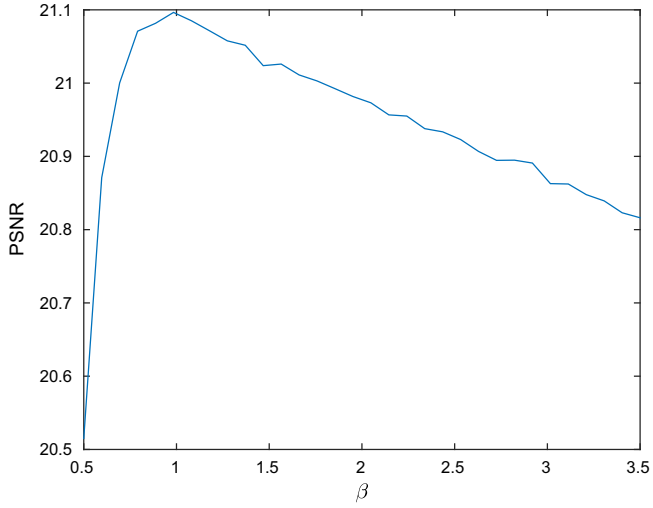


Fig. 2. Average PSNR on eight test images as a function of β for peak = 0.5.

3.1.2. Poisson deblurring

When dealing with the deblurring problem, H represents a blur matrix. The first P&PP step is no longer separable and usually no analytical solution is available. However the problem is convex and a common way to solve it is by using iterative optimization methods, which usually require the gradient. Turning back to our problem, the gradient of $L_\lambda(x)$ is given by

$$\nabla_x L_\lambda = -H^T(y/(Hx)) + H^T 1 + \lambda(x - v + u). \tag{17}$$

where "/" stands for element-wise division. As can be seen, this gradient is easy to compute, as it requires blurring the temporary solution x, the constant vector 1 and the vector $y/(Hx)$ in each such computation.

3.2. Details and improvements

The focus was given to two different inverse problems - the denoising and deblurring problem scenarios. In both, BM3D was chosen as the Gaussian denoiser, as it provides very good results and has an efficient implementation. Also, in the denoising scenario the multiple prior M-P^{4}IP algorithm was tested. As a second denoiser a simplified version of [27] was chosen, which is a multi-

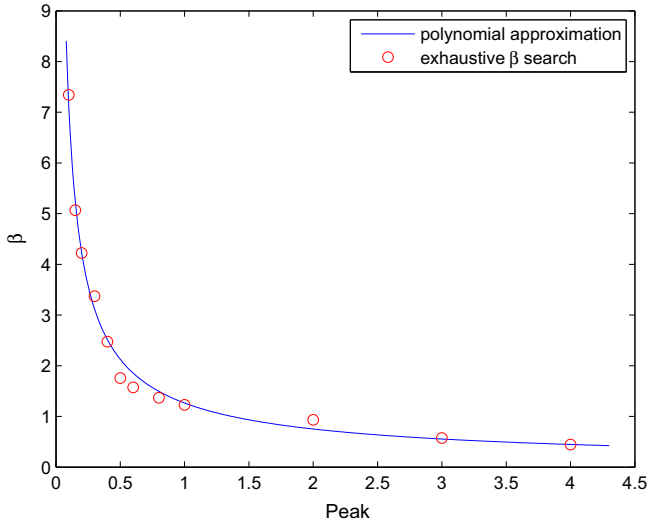


Fig. 3. Optimal β and polynomially approximated β as a function of the peak.

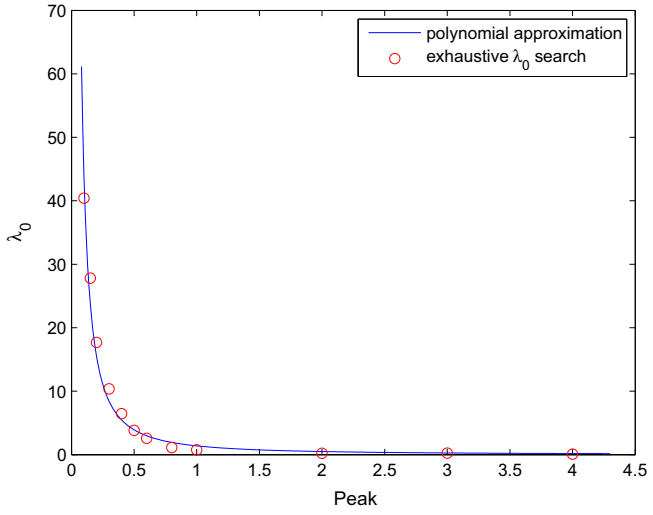


Fig. 4. Optimal λ_0 and polynomially approximated λ_0 as a function of the peak.

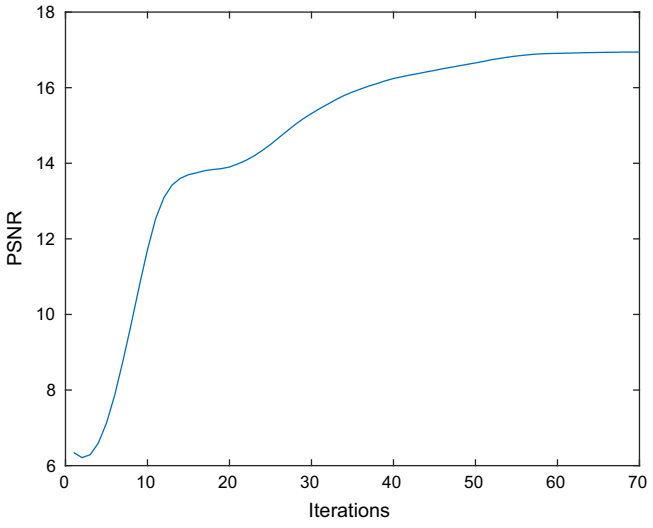


Fig. 5. PSNR value as a function of the iteration of the image Cameraman for the denoising experiment.

Table 1
Denoising without binning PSNR values [dB].

Method	Peak	Saturn	Flag	Camera	House	Swoosh	Peppers	Bridge	Ridges	Average
BM3D	0.1	19.42	13.05	15.66	16.28	16.93	15.61	15.68	**20.06**	16.59
SPDA		17.40	**13.35**	14.36	14.84	15.12	14.28	14.60	19.86	15.48
P^{4}IP		**21.14**	13.18	**16.93**	**18.34**	**21.22**	**16.11**	**16.43**	18.66	**17.75**
BM3D	0.2	22.02	14.28	17.35	18.37	19.95	17.10	17.09	21.27	18.43
SPDA		21.52	**16.58**	16.93	17.83	18.91	16.75	16.80	**23.25**	18.57
P^{4}IP		**23.01**	14.98	**17.81**	**19.48**	**23.56**	**17.18**	**17.50**	21.08	**19.32**
BM3D	0.5	23.86	15.87	18.83	20.27	22.92	18.49	18.24	23.37	20.23
SPDA		**25.50**	**19.67**	18.90	20.51	24.21	18.66	18.46	**27.76**	**21.71**
P^{4}IP		25.07	16.31	**19.20**	**20.92**	**25.72**	**18.74**	**18.50**	24.25	21.09
BM3D	1	25.89	18.31	20.37	22.35	26.07	19.89	19.22	26.26	22.30
SPDA		27.02	**22.54**	20.23	**22.73**	26.28	19.99	19.20	**30.93**	**23.61**
P^{4}IP		**27.11**	18.89	**20.48**	22.72	**27.82**	**20.72**	**19.28**	27.25	23.03
BM3D	2	27.42	20.81	**22.13**	24.18	28.09	21.97	**20.31**	29.82	24.34
SPDA		**29.38**	**24.92**	21.54	**25.09**	29.27	21.23	20.15	**33.40**	**25.62**
P^{4}IP		28.89	20.98	21.95	24.64	**29.54**	**22.33**	20.23	30.47	24.88
BM3D	4	29.40	23.04	**23.94**	26.04	30.72	**24.07**	**21.50**	32.39	26.39
SPDA		**31.04**	**26.27**	21.90	26.09	**33.20**	22.09	20.55	**36.05**	**27.15**
P^{4}IP		30.83	22.29	23.33	**26.35**	31.67	23.90	21.12	32.86	26.54

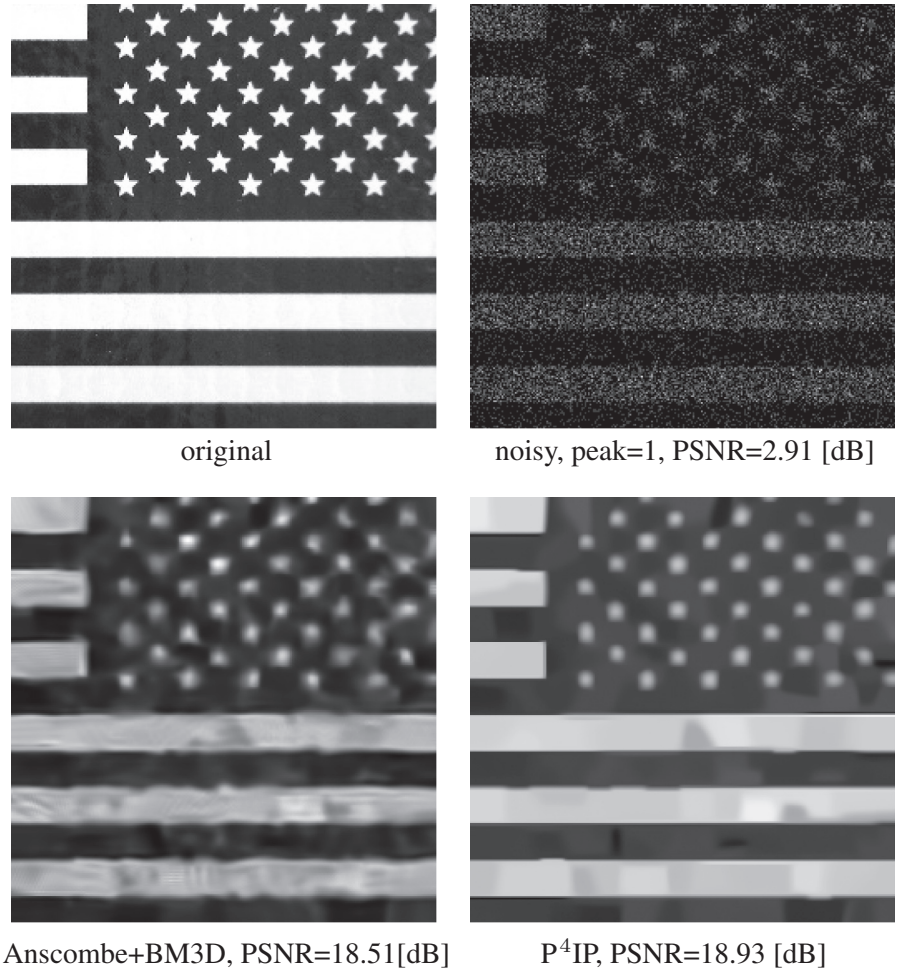


Fig. 6. The image Flag with peak 1 - denoising (no binning) results.

scale denoising algorithm. Multi-scale considerations are not used in BM3D, and therefore the two algorithms joint together may form a more powerful denoising prior.

In the low SNR case, an improvement in the recovery can be achieved by using a technique called binning: The noisy image is down scaled and the algorithm is applied on the smaller sized image that has a better SNR, since the photon count of the merged pixels are added up. Once the final result of the algorithm is obtained, up-scaling is applied by a simple linear interpolation. This technique leads to better results, and also reduces runtime as the operation is done on smaller images. All the experiments reported below with binning use a 3:1 shown-scaling in each axis.

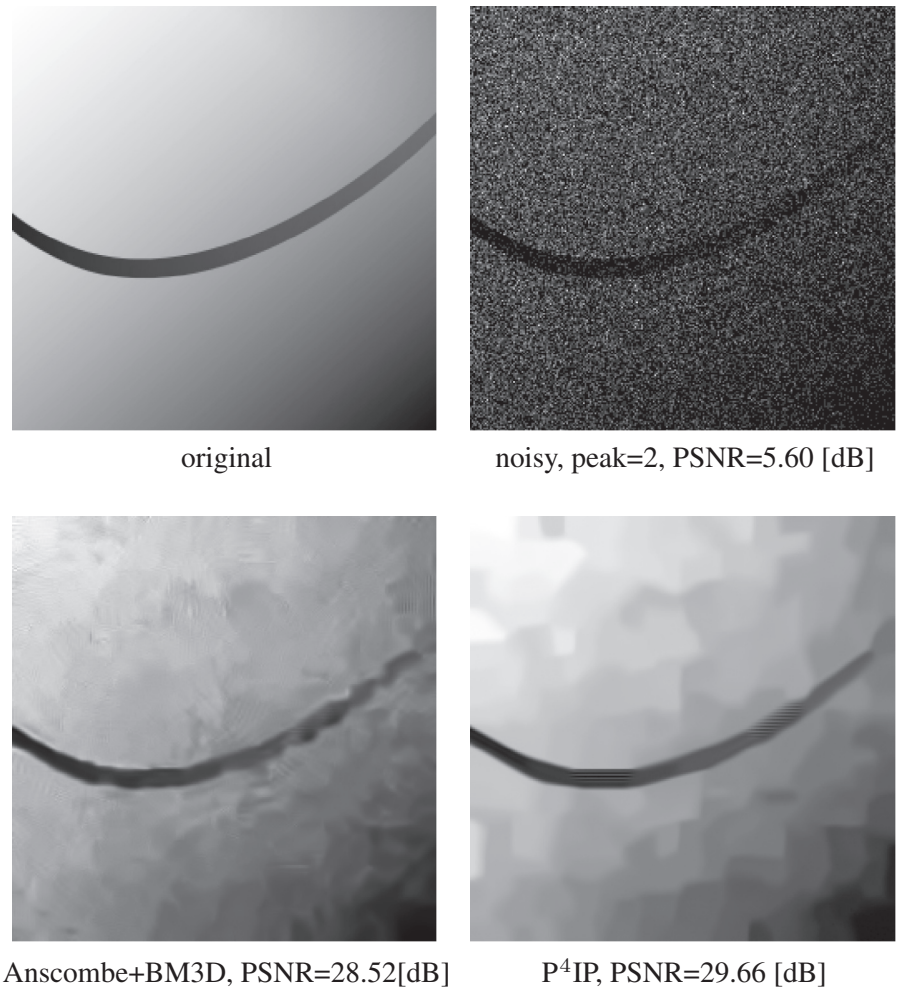


Fig. 7. Peak 2 - denoising (no binning) results.

Binning can only be used in the denoising scenario as down sampling doesn't commute with the blur operator. Moreover, using binning on deblurring tasks may be viewed as merely an approximation because the blur kernel undergoes down-sampling too. Another problem with binning which holds true for denoising, but has much stronger effect in deblurring, happens in the upscaling stage, where artifacts from the restoration algorithm are amplified during this process, because the down-sampled deblurred image suffers from them more than the simpler-to-produce denoised one. Thus using binning in the deblurring scenario leads to sub optimal reconstruction. Obviously, binning could be used in the deblurring scenario by choosing down sampling as the forward operator H.

As mentioned before, when dealing with the deblurring scenario, the first P&PP step cannot be solved analytically, and the minimization is done using convex optimization tools. L-BFGS [24] is a good choice for this task. In order to avoid calculating $\ln(\cdot)$ where Hx is negative, it is preferred to optimized the surrogate function

$$f(x) = \begin{cases} L_\lambda(x) & x < \varepsilon \\ ax^2 + bx + c & x \geqslant \varepsilon \end{cases} \tag{18}$$

where the coefficients a, b and c were chosen such that this function and its derivative coincides with L_λ and its derivative at $x = \epsilon$. As $x \to 0$ we get that $Hx \to 0$ and $L_\lambda(x) \to \inf$, therefore choosing a small enough ϵ value guarantees that the surrogate function will have the same minimum as L_λ and all entries in Hx are positive. The chosen value for ϵ is 10^{-10}.

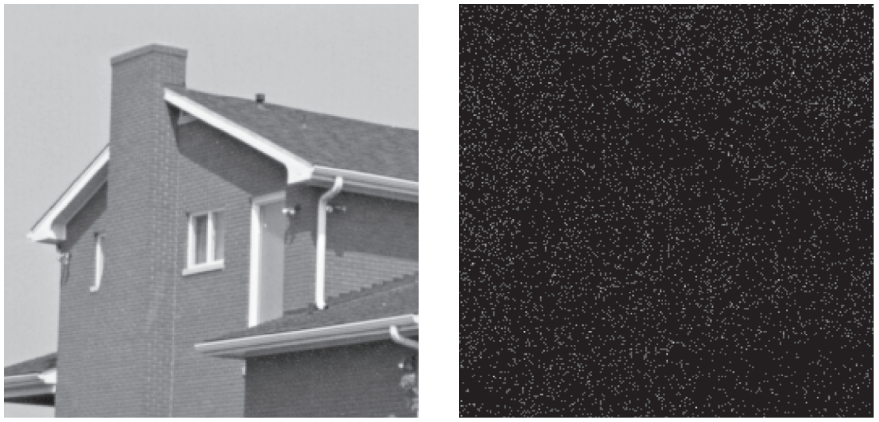
3.2.1. Choice of parameters

Both scenarios required appropriate choice of parameters and their values. The first important parameter is β - the prior weight. If the noise is very weak, β should be small because the noisy image has to be only slightly modified. However, when the noise level is high, the restoration process should rely more on the prior knowledge and therefore β should be large. Empirical results show that inappropriate choice of β leads to poor results, sometimes by several dB. Fig. 2 shows the average PSNR for peak 0.5 on eight test images as a function of β around β's optimal value.

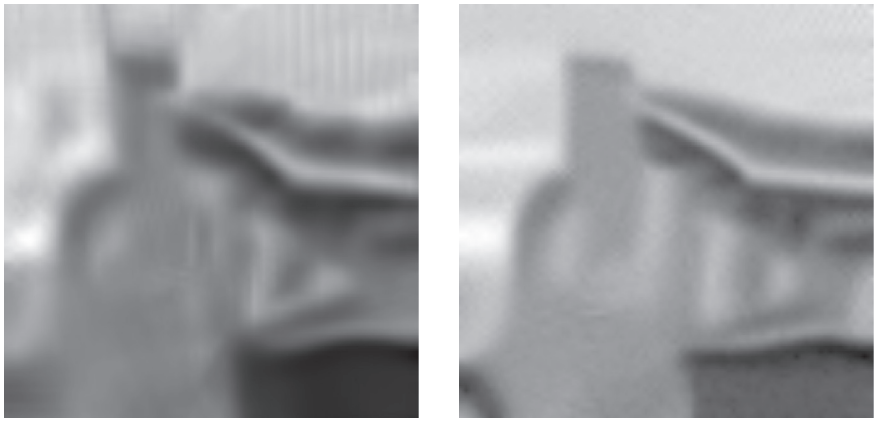
Interestingly, optimal β and λ_0 can be approximated by simple polynomials. Figs. 3 and 4 show the optimal β and λ_0 and their polynomial approximation as a function of the peak. The optimal β and λ were found by a joint exhaustive search.

Another two parameters that have big effect on the reconstruction relate to the choice of λ. This parameter is considered to be proportional to the inverse of the step size. We found that increasing λ at each iteration gives better results then a constant λ. Thus, λ update requires two parameters. The first is λ_0 - the initial value of λ, and the second, λ_{step} - the value λ is multiplied by at each iteration.

Another important parameter is the number of iterations. A fixed number of iterations was chosen, but of course this parameter can also be learned or even estimated, similarly to what is done in [22,4]. For denoising this value was set to 50 when binning was



original (a) noisy, peak=0.2, PSNR=-4.10 [dB]



Anscombe+BM3D, PSNR=19.90[dB] (b) M-P^{4}IP, PSNR=20.18 [dB]

Fig. 8. Peak 0.2 denoising (with binning).

used, and to 70 without it. M-P^{4}IP was tested only with binning and the number of iteration is set to 47. In the deblurring scenario 60 iterations are used. Setting the number of iterations is important as the PSNR starts decreasing after a while. This happens because the objective in Eq. (5) which is minimized, is only an approximation to the PSNR. Thus, the maximum of the PSNR is achieved near the minimum of this penalty function. This suggests that the optimal iteration number is a finite value, which should be chosen wisely.

For a given noise peak, each parameter was tuned on a series of 8 images. Out of each original image five degraded images were generated, noisy for the denoising scenario, and blurred and noisy for the deblurring scenario. Multiple peak values in the range of 0.2–4 were tested. Optimal λ_0 and β were found to have strong correlation with the peak value. On the other hand, λ_{step} has a weak dependence on the peak, and was thus chosen independently of it. For all scenarios λ_{step} was set to 1.065 without binning and to 1.1 with it. We observed that the initialization of v^0 does not lead to a noticeable change in the final reconstruction, and thus it is set to 0.

4. Experiments

4.1. Denoising

The algorithm was tested for peak values 0.1, 0.2, 0.5, 1, 2 and 4. To evaluate our algorithm it was compared to BM3D with the refined inverse Anscombe transform [19], and to [13], which leads to the best of our knowledge, to state of the art results. All algorithms were tested with and without binning. The results are shown in Table 1, where each value is the average of five noisy realizations. While the reported results refer to BM3D, other options were tested as well (e.g. K-SVD [10], WNNM [14]) and led to similar results in spirit and roughly speaking, for the same parameter setting strategy. BM3D's ones were reported simply because they were the best in terms of performance vs. computation trade-off.

Figs. 6–9 show several such results. Here each PSNR value refers to specific noise realization. As can be seen, the propose approach competes favorably with the BM3D+Anscombe and state-of-the-art algorithms. Binning is found to be beneficial for all algorithms when dealing with low peak values. As for run-times, our algorithm takes roughly 10 s/image when binning is used, nearly 0.2 s took for a single 2nd step run, and negligible time (less then one millisecond) for steps 1 and 3. This should be compared to the BM3D +Anscombe that runs faster (0.2 s/image), and the SPDA method [9] which is much slower (an average of 15–20 min/image). When removing the binning, our algorithm runs for about one minute, with 0.6 s on average for a single 2nd step run. BM3D+Anscombe run takes one second and SPDA runs for approximately 10 h. These runtime evaluations were measured on an i7 with 8 GB RAM laptop. Practical convergence of the algorithm on the image "Cameraman" with noise peak 0.1 is shown in Fig. 5. Due to the modification of the value of λ along the iterations, we get the non-consistent curve as seen in Fig. 5. Nevertheless, this approach does lead to improved results over the alternative (and more consistent) fixed λ option.

As mentioned before, to check the effectivity M-P^{4}IP a combination of BM3D with a simplified version of [27] was chosen, with

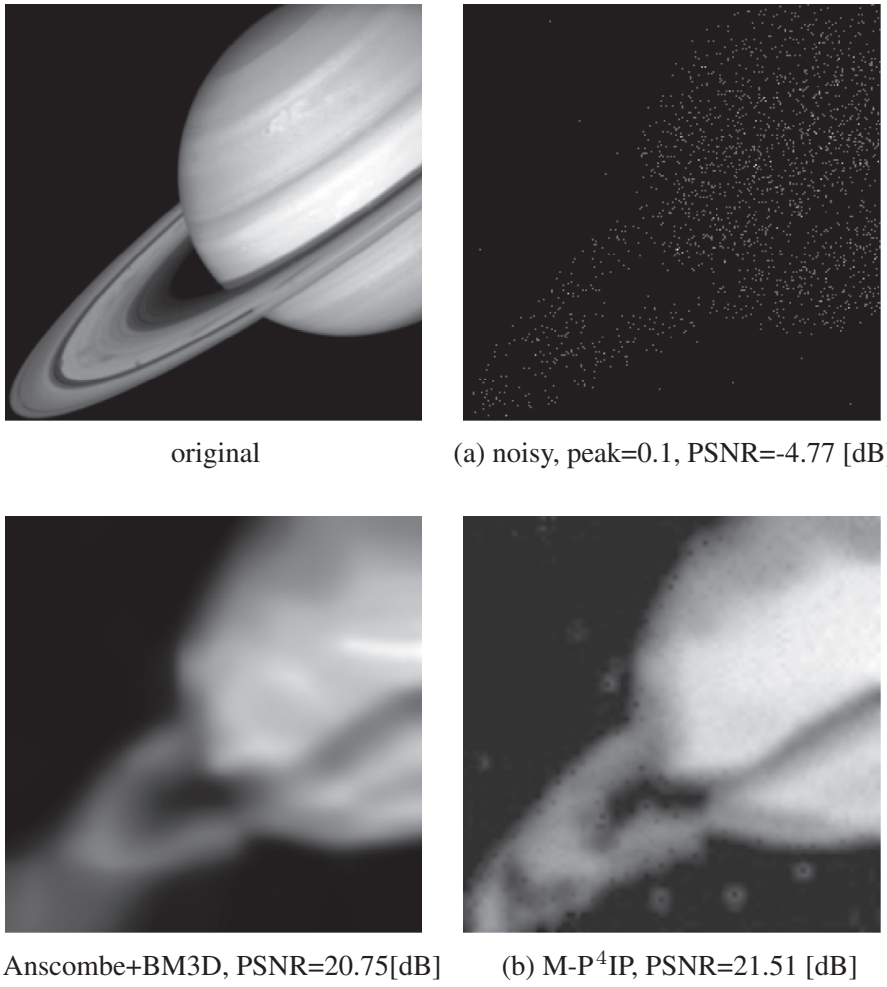


Fig. 9. Peak 0.1 denoising (with binning).

Table 2
Denoising with binning PSNR values [dB].

Method	Peak	Saturn	Flag	Camera	House	Swoosh	Peppers	Bridge	Ridges	Average
BM3Dbin	0.1	21.19	14.23	16.91	18.62	21.90	15.92	16.91	20.40	18.26
SPDAbin		22.00	**15.50**	16.75	**18.73**	21.90	16.27	**16.99**	**25.32**	**19.17**
P^{4}IP bin		**22.14**	15.03	17.16	18.55	21.88	16.24	16.79	21.85	18.70
M-P^{4}IP bin		21.83	14.87	**17.44**	18.54	**21.92**	**16.33**	16.61	22.67	18.78
BM3Dbin	0.2	23.20	16.28	18.25	19.71	24.25	17.44	17.70	23.92	20.09
SPDAbin		23.99	**18.26**	17.95	19.62	23.53	17.59	**17.82**	**27.22**	**20.75**
P^{4}IP bin		23.92	17.17	18.36	19.86	24.64	17.39	17.63	24.52	20.44
M-P^{4}IP bin		**24.09**	16.49	**18.52**	**19.94**	**25.00**	**17.63**	17.70	24.56	20.49
BM3D bin	0.5	25.70	18.40	19.64	21.71	26.33	**19.01**	**18.67**	28.23	22.21
SPDA bin		25.83	**19.22**	18.97	21.15	**26.57**	18.63	18.57	**30.97**	**22.49**
P^{4}IP bin		**26.12**	18.19	**19.72**	21.67	26.43	18.88	18.65	27.76	22.18
M-P^{4}IP bin		**26.12**	18.15	19.71	**21.74**	26.54	18.93	18.60	28.03	22.23

suitable prior weight parameter. Peaks 0.1, 0.2 and 0.5 were tested. The results are shown in Table 2. For the tested peaks, the multiple prior algorithm show improvement. It is important to note that it was harder to find good parameters and therefore it is reasonable to expect that it is possible to improve even more.

4.2. Deblurring

In this scenario, the algorithm was tested for the peak values 1, 2 and 4 of an image that was blurred by one of the following blur kernels:

(i) a Gaussian kernel of size 25 by 25 with $\sigma = 1.6$
(ii) $\frac{1}{(1+x_1^2+x_2^2)}$ for $x_1, x_2 = -7, \ldots, 7$
(iii) 9×9 uniform

To evaluate our algorithm it was compared to IDD-BM3D [7] with the refined inverse Anscombe transform [19]. It is important to note that such a scheme is essentially inaccurate, because after the Anscombe operation the image cannot be considered as one blurred by the used kernel because the blur is no longer translation invariant. Although this leads to sub-optimal results, this reconstruction strategy outperforms existing state-of-the-art direct

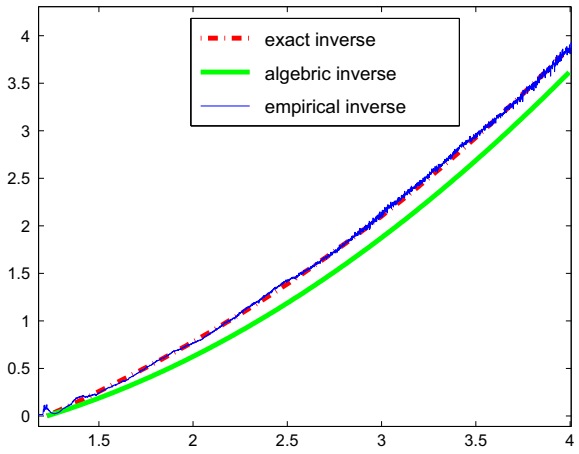


Fig. 10. Inverse VST curves.

approaches. In addition, the inversion part is biased, however this issue can be resolves. We empirically searched for the best pixel-wise transform by learning the best inverse curve. This was done by constructing a graph, in which its x axis values are pixels after the IDD-BM3D step and its y axis are values of the original pixels. Such a graph was constructed for several images at various peak levels. The results are shown in Fig. 10. This empirical curve coincides with the refined inverse, even though the tested scenario is deblurring and not denoising. Such a result implies that the refined inverse transform is useful in the deblurring task just as well. We further note that a transform that takes into account all pixels that are effected by the blur kernel, could potentially outperform the proposed VST scheme, however, such a transform is harder to compute. In addition such a transform would be dependent on the blur kernel. Also, the obtained results were compared to Ma et al. [17] which produces state of the art results. The work reported in [17] requires an appropriate parameter λ, and was thus set for each peak, such that it produces the highest average PSNR on all tested images.

The results are shown in Tables 3–5. Figs. 12–14 show specific results to better assess the visual quality of the outcome.

It is clearly shown that in this scenario P^{4}IP outperforms the Anscombe-transform framework. Practical convergence of the algorithm on the image "Cameraman" with noise peak 1 is shown in Fig. 11.

The runtime for a single image took about 2–3 min, with 1.5 s for a single 1st step run and with 0.7 s on average for a single 2nd step run. The 3rd step runtime is negligible. The runtime was measured on an i7, 8G RAM laptop, about as fast as the Anscombe transform based algorithm. Although the noise is strong,

Table 3
Deblurring PSNR values for blur kernel (i) [dB].

Method	Peak	Saturn	Flag	Camera	House	Swoosh	Peppers	Bridge	Ridges	Average
BM3D	1	24.32	16.18	19.39	21.06	**26.51**	18.47	18.34	22.06	20.79
Ma et al.		24.17	15.25	18.92	20.69	22.82	18.92	18.56	23.64	20.37
P^{4}IP		**25.69**	**17.97**	**19.84**	**21.93**	**26.51**	**19.48**	**19.03**	**25.56**	**22.00**
BM3D	2	**26.07**	17.78	20.61	22.66	28.61	19.84	19.28	25.71	22.57
Ma et al.		25.47	16.43	19.84	21.86	24.39	19.96	19.39	25.43	21.60
P^{4}IP		25.95	**19.49**	**20.78**	**23.33**	**28.67**	**20.47**	**19.67**	**28.38**	**23.34**
BM3D	4	28.05	20.25	**21.66**	**24.69**	30.30	**21.25**	**20.20**	29.05	24.43
Ma et al.		27.01	17.49	20.56	22.83	25.52	20.86	20.13	27.15	22.69
P^{4}IP		**28.81**	**20.44**	21.37	24.51	**30.62**	21.11	20.13	**31.42**	**24.80**

Table 4
Deblurring PSNR values for blur kernel (ii) [dB].

Method	Peak	Saturn	Flag	Camera	House	Swoosh	Peppers	Bridge	Ridges	Average
BM3D	1	24.36	15.53	18.99	20.81	25.83	18.24	18.20	21.21	20.40
Ma et al.		23.51	15.24	18.84	20.41	22.13	18.81	18.30	22.97	20.03
P^{4}IP		**25.14**	**17.07**	**19.50**	**21.52**	**25.89**	**19.05**	**18.69**	**24.28**	**21.39**
BM3D	2	26.02	16.58	20.01	22.15	**28.33**	19.29	18.98	24.38	21.97
Ma et al.		25.07	16.62	19.88	21.66	23.63	**19.98**	**19.38**	24.90	21.39
P^{4}IP		**26.39**	**18.61**	**20.18**	**22.49**	28.29	19.80	19.25	**26.63**	**22.70**
BM3D	4	27.64	19.00	**20.84**	**23.68**	29.45	20.55	19.71	27.52	23.55
Ma et al.		26.42	17.83	20.73	22.73	24.67	**21.05**	**20.28**	26.35	22.51
P^{4}IP		**28.48**	**19.80**	20.76	23.58	**29.70**	20.56	19.70	**29.20**	**23.97**

Table 5
Deblurring PSNR values for blur kernel (iii) [dB].

Method	Peak	Saturn	Flag	Camera	House	Swoosh	Peppers	Bridge	Ridges	Average
BM3D	1	24.11	15.46	18.93	20.71	**26.23**	18.12	18.17	21.48	20.40
Ma et al.		24.27	14.86	18.81	20.64	23.32	18.73	18.38	23.29	20.29
P^{4}IP		**24.36**	**17.12**	**19.49**	**21.37**	26.03	**19.04**	**18.64**	**23.53**	**21.20**
BM3D	2	**26.06**	16.54	19.93	22.20	**28.26**	19.29	18.83	24.69	21.97
Ma et al.		25.56	15.82	19.44	21.63	24.84	19.57	19.03	24.94	21.35
P^{4}IP		25.62	**18.61**	**20.11**	**22.54**	28.17	**19.81**	**19.19**	**25.83**	**22.48**
BM3D	4	27.41	18.83	20.63	**23.47**	29.81	20.36	19.63	27.56	23.46
Ma et al.		26.82	16.66	19.99	22.35	25.97	20.27	19.65	26.43	22.27
P^{4}IP		**27.97**	**19.77**	**20.66**	23.39	**29.93**	**20.47**	**19.71**	**29.15**	**23.88**

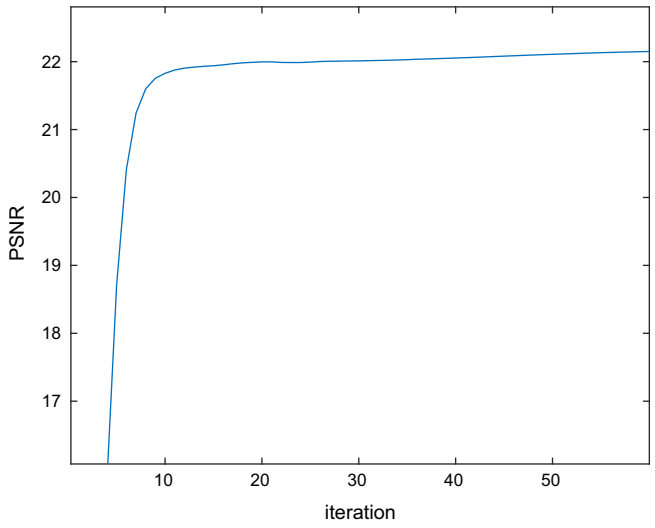


Fig. 11. PSNR value as a function of the iteration of the image Cameraman for the deblurring experiment.

the blurring of the images adds degradation. Simply applying the denoising version of P^{4}IP leads to inferior results compared to the deblurring version by an average of 0.4 dB for peak 1, 0.8 dB for peak 2 and 1.7 dB for peak 4. This implies that the first P^{4}IP step which deals with the specific inverse problem should be chosen correctly.

5. Conclusion and discussion

This work proposes a new way to integrate Gaussian denoising algorithms to Poisson noise inverse problems, by using the Plug-and-Play framework, this way taking advantage of the existing Gaussian solvers. The integration is done by simply using the Gaussian denoiser as a "black box" as part of the overall algorithm. This work demonstrates this paradigm on two problems - image denoising and image deblurring. Numerical results show that our algorithm outperforms the Anscombe-transform based framework in lower peaks, and competes favorably with it on other cases. These results could be further improved by using the proposed extension of Plug-and-Play, which enables to combine multiple Gaussian denoising algorithms.

Further work should be done in order to better tune the algorithm's parameters, similar to [8]. Also, there exist many techniques that may improve the obtained results. For instance, averaging the final results of several algorithm runs with slightly different parameters may be beneficial. Of course, this comes at cost of run time. It is also interesting to learn more closely the rela-

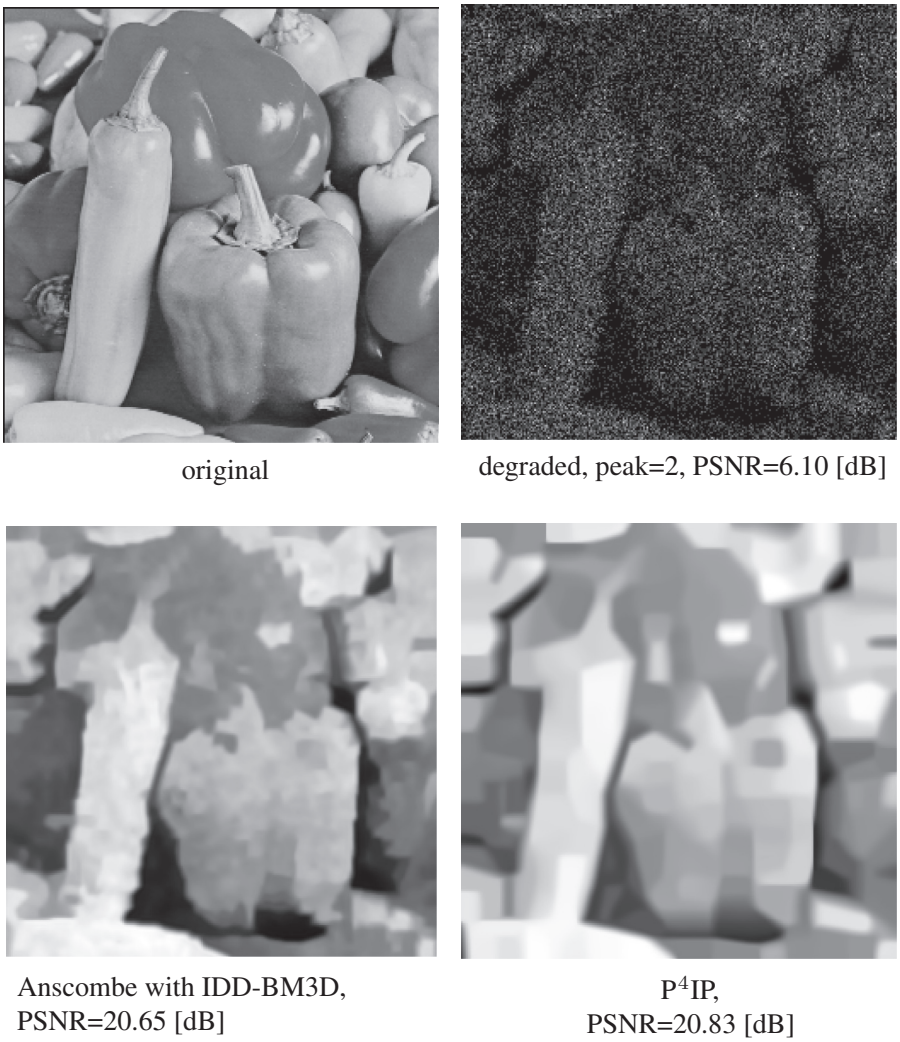


Fig. 12. The image Peppers with peak 2 and blur kernel (i) - deblurring results.

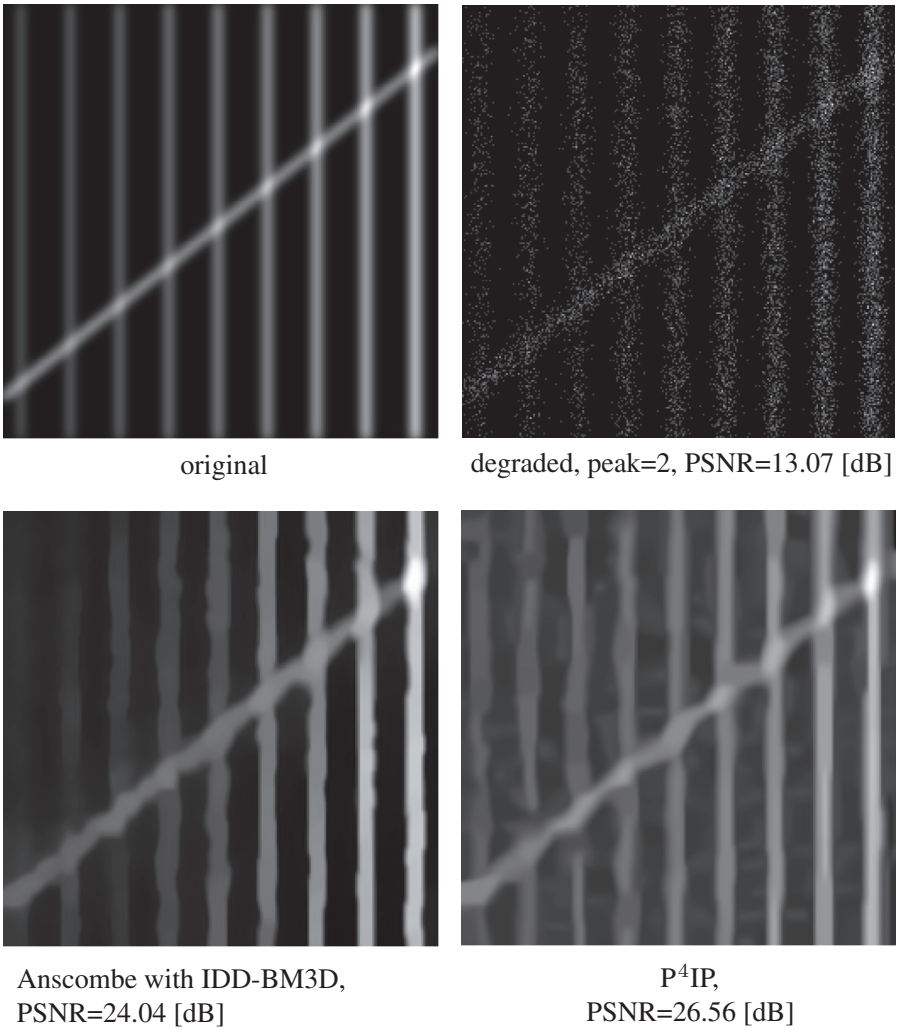


Fig. 13. The image Ridges with peak 2 and blur kernel (ii) - deblurring results.

tion between the Anscombe transform-based framework and ours. We have found that under certain initialization conditions, in the first step P^{4}IP does variance stabilization that is as good as Anscombe's one. It would be possible that more could be said about this matter.

Different noise models are also treatable by a similar scheme. For instance, when dealing with Poisson- Gaussian noise, which is common in CCD sensors, $l(x)$ is given in [15]. It would be interesting to see the effectiveness of the P&PP approach for this case.

Appendix A. Derivation of first denoising ADMM step

In the denoising case $H = I$ and we get that $l(x)$ is given by

$$l(X) = -y^T \ln(x) + 1^T \ln(\Gamma(y+1)) + 1^T x. \tag{19}$$

The augmented Lagrangian is thus

$$L_\lambda = -y^T \ln(x) + 1^T x - \beta \ln(P(v)) + \frac{\lambda}{2}\|x - v + u\| - \frac{\lambda}{2}\|u\|_2^2, \tag{20}$$

and the first ADMM step becomes

$$\begin{aligned} x^{k+1} &= \arg\min_x \; L_\lambda(x, v^k, u^k) \\ &= \arg\min_x -y^T \ln(x) + 1^T x + \frac{\lambda}{2}\left\|x - v^k + u^k\right\|_2^2. \end{aligned} \tag{21}$$

The first step (x update) is a convex and separable, implying that each entry of x can be treated separately. Furthermore, computing the elements of x is easily handled leading to a closed form expression. By differentiating L_λ by $x[i]$ and equating to 0 we get

$$-\frac{y[i]}{x[i]} + 1 + \lambda(x[i] - v^k[i] + u^k[i]) = 0. \tag{22}$$

Thus, we get that

$$x[i] = \frac{(\lambda(v^k[i] - u^k[i]) - 1) + \sqrt{(\lambda(v^k[i] - u^k[i]) - 1)^2 + 4\lambda y[i]}}{2\lambda}. \tag{23}$$

As y is non negative, the expression inside the square root is also non negative and causes the resulted x to be non negative also. Another possible solution could have been the second root of Eq. (22), but this solution is purely negative and thus uninformative.

Appendix B. Derivation of multiple P&PP

We start by looking at the general ADMM formulation

$$\begin{aligned} &\arg\min \; l(x) + s(v) \\ &s.t.\; Ax - Bv = c. \end{aligned} \tag{24}$$

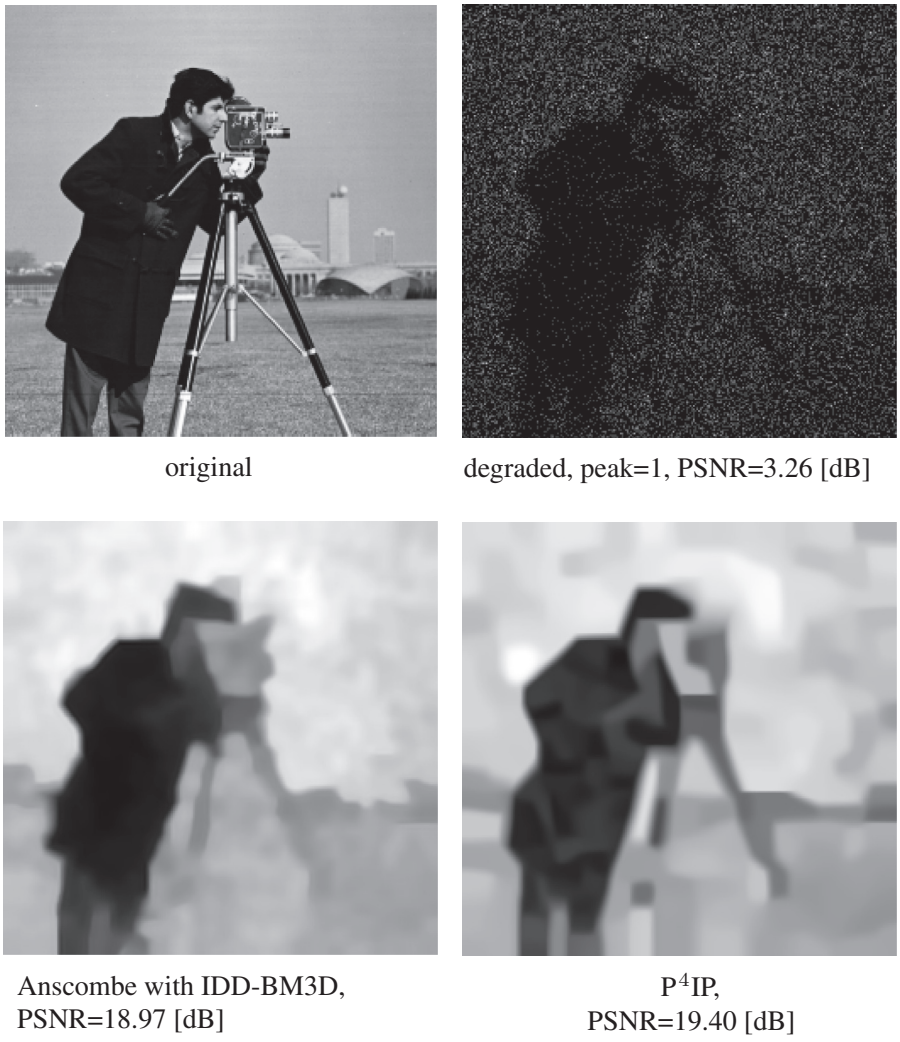


Fig. 14. The image Cameraman with peak 1 and blur kernel (iii) - deblurring results.

The augmented Lagrangian is:

$$L_\lambda = l(x) + s(v) + \frac{\lambda}{2}\|Ax - Bv - c + u\|_2^2 - \frac{\lambda}{2}\|u\|_2^2. \tag{25}$$

Choosing appropriate matrices A and B, and using $s(\cdot)$ as a function of two separable priors leads to the multiple prior version of P&PP. Let $A = \begin{bmatrix} I \\ I \end{bmatrix}$, $B = I$, $c = 0$, $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$, and $s(v) = \beta_1 s_1(v_1) + \beta_2 s_2(v_2)$. By plugging the above into Eq. (24) we have:

$$\begin{aligned} &\arg\min\ l(x) + \beta_1 s_1(v_1) + \beta_2 s_2(v_2) \\ &s.t.\ x = v_1, \quad x = v_2. \end{aligned} \tag{26}$$

The augmented Lagrangian is:

$$\begin{aligned} L_\lambda = {} & l(x) + \beta_1 s_1(v_1) + \beta_2 s_2(v_2) + \frac{\lambda}{2}\|x - v_1 + u_1\|_2^2 \\ & + \frac{\lambda}{2}\|x - v_2 + u_2\|_2^2 - \frac{\lambda}{2}\|u_1\|_2^2 - \frac{\lambda}{2}\|u_2\|_2^2, \end{aligned} \tag{27}$$

where $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$, which leads to the multiple version of the P&PP.

References

[1] F.J. Anscombe, The transformation of Poisson, binomial and negative-binomial data, Biometrika (1948) 246–254.
[2] J. Boulanger, C. Kervrann, P. Bouthemy, P. Elbau, J.-B. Sibarita, J. Salamero, Patch-based nonlocal functional for denoising fluorescence microscopy image sequences, IEEE Trans. Med. Imag. 29 (2) (2010) 442–454.
[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, Found. Trends® Mach. Learn. 3 (1) (2011) 1–122.
[4] M. Carlavan, L. Blanc-Féraud, Sparse Poisson noisy image deblurring, IEEE Trans. Image Process. 21 (4) (2012) 1834–1846.
[5] K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, Image denoising by sparse 3-d transform-domain collaborative filtering, IEEE Trans. Image Process. 16 (8) (2007) 2080–2095.
[6] A. Danielyan, V. Katkovnik, K. Egiazarian, Image deblurring by augmented Lagrangian with bm3d frame prior, in: Workshop on Information Theoretic Methods in Science and Engineering (WITMSE), Tampere, Finland, 2010, pp. 16–18.
[7] A. Danielyan, V. Katkovnik, K. Egiazarian, Bm3d frames and variational image deblurring, IEEE Trans. Image Process. 21 (4) (2012) 1715–1728.
[8] C.-A. Deledalle, F. Tupin, L. Denis, Poisson NL means: unsupervised non local means for Poisson noise, in: 2010 17th IEEE International Conference on Image Processing (ICIP), IEEE, 2010, pp. 801–804.
[9] F.-X. Dupé, M. Fadili, J.-L. Starck, Deconvolution under Poisson noise using exact data fidelity and synthesis or analysis sparsity priors, Stat. Methodol. 9 (1) (2012) 4–18.
[10] M. Elad, M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, IEEE Trans. Image Process. 15 (12) (2006) 3736–3745.

[11] M.A. Figueiredo, J.M. Bioucas-Dias, Restoration of Poissonian images using alternating direction optimization, IEEE Trans. Image Process. 19 (12) (2010) 3133–3145.
[12] M. Fisz, The limiting distribution of a function of two independent random variables and its statistical application, Colloquium Mathematicae, vol. 3, Institute of Mathematics Polish Academy of Sciences, 1955, pp. 138–146.
[13] R. Giryes, M. Elad, Sparsity-based Poisson denoising with dictionary learning, IEEE Trans. Image Process. 23 (12) (2014) 5057–5069.
[14] S. Gu, L. Zhang, W. Zuo, X. Feng, Weighted nuclear norm minimization with application to image denoising, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 2862–2869.
[15] A. Jezierska, E. Chouzenoux, J.-C. Pesquet, H. Talbot, A primal-dual proximal splitting approach for restoring data corrupted with Poisson-Gaussian noise, in: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2012, pp. 1085–1088.
[16] M.R. Keenan, P.G. Kotula, Accounting for Poisson noise in the multivariate analysis of ToF-SIMS spectrum images, Surf. Interface Anal. 36 (3) (2004) 203–212.
[17] L. Ma, L. Moisan, J. Yu, T. Zeng, A dictionary learning approach for Poisson image deblurring, IEEE Trans. Med. Imag. 32 (7) (2013) 1277–1289.
[18] J. Mairal, F. Bach, J. Ponce, G. Sapiro, A. Zisserman, Non-local sparse models for image restoration, in: 2009 IEEE 12th International Conference on Computer Vision, IEEE, 2009, pp. 2272–2279.
[19] M. Makitalo, A. Foi, Optimal inversion of the Anscombe transformation in low-count Poisson image denoising, IEEE Trans. Image Process. 20 (1) (2011) 99–109.
[20] N. Pustelnik, C. Chaux, J.-C. Pesquet, Parallel proximal algorithm for image restoration using hybrid regularization, IEEE Trans. Image Process. 20 (9) (2011) 2450–2462.
[21] I. Rodrigues, J. Sanches, J. Bioucas-Dias, Denoising of medical images corrupted by Poisson noise, in: ICIP 2008. 15th IEEE International Conference on Image Processing, 2008, IEEE, 2008, pp. 1756–1759.
[22] Y. Romano, M. Elad, Improving K-SVD denoising by post-processing its method-noise, in: ICIP, 2013, pp. 435–439.
[23] J. Salmon, Z. Harmany, C.-A. Deledalle, R. Willett, Poisson noise reduction with non-local PCA, J. Math. Imag. Vision 48 (2) (2014) 279–294.
[24] M. Schmidt. minfunc: Unconstrained Differentiable Multivariate Optimization in Matlab, 2005. <http://www.cs.ubc.ca/schmidtm/Software/minFunc.html>.
[25] J. Schmitt, J. Starck, J. Casandjian, J. Fadili, I. Grenier, Poisson denoising on the sphere: application to the Fermi gamma ray space telescope, Astron. Astrophys. 517 (2010) A26.
[26] S. Sreehari, S. Venkatakrishnan, B. Wohlberg, L.F. Drummy, J.P. Simmons, C.A. Bouman, Plug-and-play Priors for Bright Field Electron Tomography and Sparse Interpolation, 2015. arXiv preprint 1512.07331.
[27] J. Sulam, B. Ophir, M. Elad, Image denoising through multi-scale learnt dictionaries, in: 2014 IEEE International Conference on Image Processing (ICIP), IEEE, 2014, pp. 808–812.
[28] S.V. Venkatakrishnan, C.A. Bouman, B. Wohlberg, Plug-and-play priors for model based reconstruction, in: 2013 IEEE Global Conference on Signal and Information Processing (GlobalSIP), IEEE, 2013, pp. 945–948.
[29] G. Yu, G. Sapiro, S. Mallat, Solving inverse problems with piecewise linear estimators: from Gaussian mixture models to structured sparsity, IEEE Trans. Image Process. 21 (5) (2012) 2481–2499.
[30] B. Zhang, J.M. Fadili, J.L. Starck, Wavelets, ridgelets, and curvelets for Poisson noise removal, IEEE Trans. Image Process. 17 (7) (2008) 1093–1108.