Handling Poisson Inverse Problems by the Plug-and-Play Priors scheme

Arie Rond

Technion - Computer Science Department - M.Sc. Thesis MSC-2016-14 - 2016

Handling Poisson Inverse Problems by the Plug-and-Play Priors scheme

Research Thesis

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science

Arie Rond

Submitted to the Senate
of the Technion — Israel Institute of Technology
Iyar 5776 Haifa May 2016

This research was carried out under the supervision of Prof. Michael Elad, in the Faculty of Computer Science.

The generous financial help of the Technion is gratefully acknowledged.

Contents

List of Figures

Abstract

A common problem in the process of acquiring a signal is its corruption by noise. When dealing with photon counting, which is common when acquiring certain weak signals, such a noise can be modeled by the Poisson distribution, where each measurement is a Poisson random variable, with its mean equal to the clean signal value. Restoration of the clean signal out of its noisy measurement is essential in various fields, and thus effective and efficient algorithms are required for this task.

In the case of high signal-to-noise-ratio (SNR), where the noise level is low, the Poisson statistics resemble the Gaussian one. In such cases it is possible to approximately transform the Poisson contaminated signal into a Gaussian additive noise form, with a variance independent of the mean. This is done by a variance stabilizing transform (VST), such as the well known and widely used Anscombe transform. This transform is important and appealing, as it is easy to compute, and becomes handy in various inverse problems with Poisson noise contamination. The solution to such problems can be done by first applying the Anscombe transform, then applying a Gaussian-noise-oriented restoration algorithm of choice, and finally applying an inverse Anscombe transform. The attractiveness of this approach is due to the abundance of high-performance restoration algorithms designed for white additive Gaussian noise. This process is known to work well for high SNR images, where the Anscombe transform provides a nearly constant variance. When the noise level is high, the above path loses much of its effectiveness, and the common practice is to replace it with a direct treatment of the Poisson distribution. Naturally, with this we lose the ability to leverage on the vastly available solvers for Gaussian noise.

In this work we suggest a novel method for coupling Gaussian denoising algorithms to Poisson noisy inverse problems. Our proposed method is based on a recently proposed approach termed "Plug-and-Play Priors". Deploying the Plug-and-Play Prior approach to such problems leads to an iterative scheme that repeats several key steps: (i) A convex programing task of simple form that can be easily treated; (ii) A powerful Gaussian denoising algorithm of choice; and (iii) A simple update step. Such a modular method, just like the Anscombe transform based path, enables other developers to plug their own Gaussian denoising algorithms to this scheme in an easy way. While the proposed method bears some similarity to the Anscombe operation approach, it is in fact based on a different mathematical basis, which holds true for all SNR ranges. We demonstrate the effectiveness of the proposed scheme for Poisson image denoising and deblurring, showing that in both applications we outperform existing Anscombe-transform-based methods in high noise regimes and compete favorably with leading direct method.

Abbreviations and Notations

AVG.	:	Average
Alg.	:	Algorithm
SNR	:	Signal to Noise Ratio
MAP	:	Maximum a posteriori Probability
PIP	:	Poisson Inverse Problems
VST	:	Variance Stabilizing Transformation
ADMM	:	Alternating Direction Method of Multipliers
PaPP	:	Plug and Play Prior
P^{4}IP	:	Plug and Play Priors for Poisson Inverse Problems
M-P^{4}IP	:	Multiple Plug and Play Priors for Poisson Inverse Problems
μ	:	Expected value
σ	:	Standard deviation
ϕ	:	Noise operator
$P(x)$	:	probability of x
$P(y\|x)$	:	probability of y given x
$N(\mu,\sigma)$	:	Normal (Gaussian) distribution with mean μ and variance σ^2
$\delta(\cdot)$	:	Kronecker delta function
$\mathfrak{R}(\cdot)$	:	Gaussian restoration algorithm
$\mathfrak{D}_\sigma(\cdot)$	:	Gaussian denoising algorithm with parameter σ
$l(x)$	:	$-ln(P(y\|x))$
$s(x)$	:	$-ln(P(x))$
L_λ	:	Augmented Lagrangian with parameter λ

Chapter 1

Introduction

In an inverse problem we are given a degraded signal, y, and aim to recover from it the original signal, x. The mathematical relation between the two signals is broadly given by $y = \phi(Hx)$, where H is some linear degradation matrix and $\phi(\cdot)$ 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 signals, given by $P(x)$.

In this work we concentrate on the case of Poisson Inverse Problems (PIP), where $\phi(\cdot)$ stands for Poisson noise contamination. In such a model for an image, the gray levels of the pixels are viewed as Poisson distributed random variables. This model is relevant in various tasks such as very low light imaging [HN03], PET and SPECT reconstruction [RSBD08], fluorescence microscopy [BKB$^+$10], astrophysics [SSC$^+$10] and spectral imaging [KK04]. Common to all these tasks is the weak measured signal intensity, which acts as a counter of particle (e.g. photons).

A fundamental property of the Poisson noise is that the SNR of the noisy image is proportional to the square root of the original image intensity values. Therefore the peak value of an image is a useful measure for 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 a Variance Stabilized Transform (VST) that converts the Poisson distribution into an approximately unit variance Gaussian additive one, which is independent of the mean of the transformed distribution [Ans48], [FN04]. Since there are highly effective algorithms for Gaussian noise restoration (e.g. [DFKE07], [EA06], [MBP$^+$09], [YSM12], [DKE12]), such methods can be used, preceded by a VST and followed by the VST's inverse. Such an approach is taken in [MF11], [ZFS08].

When dealing with lower peaks, such transformations become less efficient, and alternative methods are required, which treat the Poisson noise directly (e.g. [GE14], [SHDW14]). In recent years this direct approach has drawn considerable attention, and seems to be very successful. In this work we propose a novel strategy for Poisson inverse

problems, which belongs to the direct approach family. The appeal of the proposed method is the fact that it offers an elegant bridge between the direct and indirect methods, as it relies too on Gaussian noise removal techniques, applied iteratively.

The proposed strategy has two advantages. First, it works well for all noise levels. Second, it enables the use of existing and well-performing Gaussian denoising algorithms. Our algorithm is derived from the *Plug and Play Prior* (PaPP) [VBW13] scheme, which we introduce and further develop. We show empirical experiments that demonstrate the power of our algorithm in treating Poisson denoising and deblurring problems.

This thesis is organized in the following way: In Chapter 2 we explain more thoroughly the concept of Poisson noise. We also discuss the MAP estimation and ADMM optimization techniques, which are topics that are used for explaining our algorithm. In Chapter 3 we introduce several existing methods for solving the Poisson denoising problem. In Chapter 4 we explain the PaPP framework, which is a key concept in our work. Then, in Chapter 4.2, we suggest an extension to the PaPP for handling several different priors, which we later use. In Chapter 5 we discuss the main algorithms developed in our work, termed P^{4}IP (Plug and Play Priors for Poisson Inverse Problems) and M-P^{4}IP (Multiple Priors Plug and Play for Poisson Inverse Problems). In Chapter 6 we present numerical experiments of our algorithm, which demonstrate that the proposed algorithm outperforms existing VST based methods and compares favorably with state of the art algorithms which are based on a direct approach restoration. Finally, in chapter 7 we summarize our work and suggest open questions and further extensions for future study.

Chapter 2

Preliminaries

2.1 Poisson Noise

Signals and images are often degraded by noise. This disturbance is naturally described by a statistical model. The most common of these noise models is the additive white Gaussian noise, which describes many physical phenomena such as thermal noise. Less common but nevertheless important is the Poisson noise that obeys a Poisson distribution: Gray levels of the noisy pixels are each an independent 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:

$$P\left(y\left[i\right] \mid x\left[i\right]\right) = \begin{cases} \frac{(x[i])^{y[i]}}{y[i]!}e^{-x[i]} & \text{if } x\left[i\right] > 0 \\ \delta\left(y\left[i\right]\right) & \text{if } x\left[i\right] = 0 \end{cases}, \tag{2.1}$$

where $\delta(\cdot)$ is the Kronecker delta function. Due to the properties of Poisson distribution, the mean μ_i and the variance σ_i^2 of the i-th pixel's gray level are equal to the original clean value $x[i]$. Consequently, the SNR is given by $\frac{\mu}{\sigma} = \sqrt{x[i]}$. Thus, if the measurement is generated from a clean low intensity signal, it will suffer from a stronger noise than a signal generated from a high intensity value. Therefore, it is common to use the peak value of an image for evaluating the level of the noise in it.

Poisson noise appears in imaging mainly when the image is generated from a measure of discrete events such as photon (or any other particle) counting. The stronger the intensity of the original image is, the more photons can reach the sensor. The weaker the intensity is, the fewer the photons that are measured and the stronger the noise will be. Poisson distribution describes this relation between the intensity and the probability of acquiring a certain number of photons. In a strong signal regime, the noise can be viewed approximately as a Gaussian additive one. However, in the case of low photon count the noise becomes more severe and the Gaussian approximation is much less accurate (see Figure 2.1). This happens in applications such as PET/SPECT [RSBD08] (where you count Positrons instead of photons), fluorescence microscopy

[BKB+10], astrophysics [SSC+10], spectral imaging [KK04] and many others. In all these applications, an effective methods for restoring the original image is required.

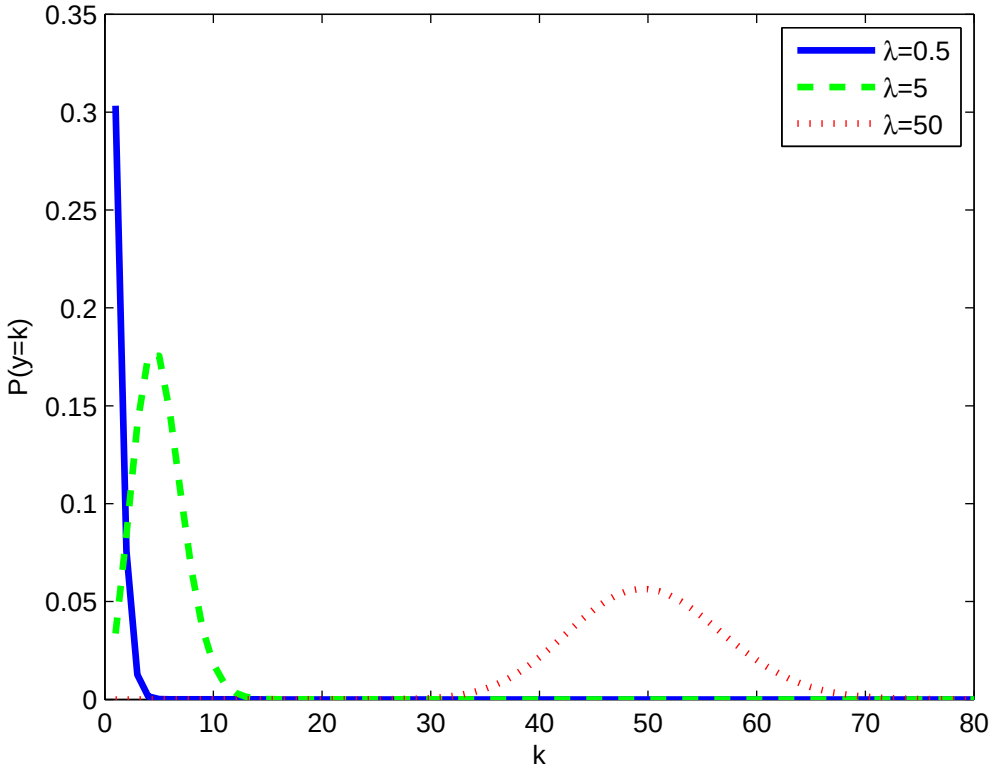


Figure 2.1: Poisson distribution for different values of λ.

2.2 MAP Estimation

When the noise statistics is known, a commonly used way to reconstruct an image based on its noisy measurements is by finding the image that has the highest probability to generate the noisy image. This can be done by maximizing the posterior probability:

$$\hat{x} = \arg\max_x P(x|y) = \arg\max_x \frac{P(y|x)\,P(x)}{P(y)} = \arg\max_x P(y|x)\,P(x)\,, \tag{2.2}$$

where y is the noisy image and $\hat{x}$ is the reconstructed one. The above objective function contains two ingredients: (i) the measurement forward model, mathematically given by the likelihood function $P(y|x)$; and (ii) a prior model for clean images, given by $P(x)$. The term $P(y|x)$ is the probability of generating a noisy image y, given the clean image, x. It is usually easy to formulate this term mathematically. Particularly, in the case that y is a Poisson noisy image of x, this term is given by Equation (2.1). The term $P(x)$ is the prior knowledge on the image x. Ideally, this term should be large for data that represent actual images, and small otherwise. In the absence of such a term, the

image reconstruction problem is often ill posed, and its solution might be none-unique. A pursuit for the appropriate prior term has drawn a lot of interest over the years, and has had a great impact on the quality of image reconstruction. Few examples for such a prior are the popular sparsity prior, where the image is assumed to be generated out of a small amount of atoms [Ela10], and the self similarity prior, where similar patches are assumed to be found abundantly in the image [BCM05].

Typically, it is more convenient to minimize the minus of the $log(\cdot)$ of Equation (2.2), which leads to the same solution achieved by maximizing Equation (2.2) directly, as $log(\cdot)$ is a monotonically increasing function. This log-likelihood minimization reads as

$$\hat{x} = \arg\min_x -\ln\left(P\left(y|x\right)\right) - \ln\left(P\left(x\right)\right). \tag{2.3}$$

The following two propositions present the MAP estimators of Gaussian and Poisson noises. These estimators will be used later in our work.

Proposition 2.2.1. *Let $y = x + n$, where x is a clean image and $n \sim N\left(0, \sigma^2 I\right)$. The MAP estimator $\hat{x}$ is given by*

$$\hat{x} = \arg\min_x \|y - x\|_2^2 - 2\sigma^2 \ln\left(P\left(x\right)\right). \tag{2.4}$$

Proof. First, note that $y \sim N\left(x, \sigma^2 I\right)$, and thus

$$P\left(y|x\right) = const \cdot exp\left(-\frac{\|y - x\|_2^2}{2\sigma^2}\right). \tag{2.5}$$

Taking -ln(·) we get

$$-\ln\left(P\left(y|x\right)\right) = const + \frac{\|y - x\|_2^2}{2\sigma^2}. \tag{2.6}$$

Plugging the above into Equation (2.3) we have

$$\hat{x} = \arg\min_x \frac{\|y - x\|_2^2}{2\sigma^2} + const - \ln\left(P\left(x\right)\right). \tag{2.7}$$

This concludes the proof as addition and multiplication by constants have no influence on the minimizer $\hat{x}$. □

Proposition 2.2.2. *Let $y \sim Pois(Hx)$, where x is a clean image and $Pois\left(Hx\right)$ is a Poisson distribution with mean Hx. The MAP estimator $\hat{x}$ is given by*

$$\hat{x} = \arg\min_x -y^T \ln(Hx) + 1^T Hx - \ln\left(P\left(x\right)\right). \tag{2.8}$$

Proof. By the Poisson distribution formula we have

$$P\left(y|x\right) = \prod_i \frac{\left(Hx\right)_i^{y_i}}{\Gamma\left(y_i + 1\right)} e^{-\left(Hx\right)_i}. \tag{2.9}$$

Taking -ln(·) on both sides leads to

$$-\ln\left(P\left(y|x\right)\right)=-\sum_{i}\ln\left(\frac{\left(Hx\right)_{i}^{y_{i}}}{\Gamma\left(y_{i}+1\right)}e^{-\left(Hx\right)_{i}}\right)=-y^{T}\ln(Hx)+1^{T}Hx+constant. \tag{2.10}$$

Plugging the above to Equation (2.3) provides

$$\hat{x}=\arg\min_{x}-y^{T}\ln(Hx)+1^{T}Hx+const-\ln\left(P\left(x\right)\right), \tag{2.11}$$

which concludes the proof. □

Remark. A Poisson denoising problem is given for the case $H=I$, and thus its MAP estimator $\hat{x}$ is given by

$$\hat{x}=\arg\min_{x}-y^{T}\ln(x)+1^{T}x-\ln\left(P\left(x\right)\right). \tag{2.12}$$

2.3 The Alternating Direction Method Of Multipliers

Consider the following problem:

$$\begin{aligned}\left(\hat{x},\hat{v}\right)=&\arg\min_{x\in R^{m},v\in R^{n}}f\left(x\right)+g\left(v\right),\\&s.t\;\;Ax+Bv=c\end{aligned} \tag{2.13}$$

for some functions f and g, matrices A and B, and a constant vector c. This formulation is common in numerous applications in many fields such as image processing and machine learning. A recent popular approach for solving Equation (2.13) is the alternating direction method of multipliers (ADMM). First,we construct an augmented Lagrangian of the form

$$L_{\lambda}=f\left(x\right)+g\left(v\right)+y^{T}(Ax+Bv-c)+\frac{\lambda}{2}\left\|Ax+Bv-c\right\|_{2}^{2}, \tag{2.14}$$

where y is the dual variable and $\lambda>0$ is called the penalty parameter. Then we iterate until convergence over the following three steps:

$$\begin{aligned}x^{k+1}&=\arg\min_{x}L_{\lambda}(x,v^{k},y^{k})\\v^{k+1}&=\arg\min_{v}L_{\lambda}(x^{k+1},v,y^{k})\\y^{k+1}&=y^{k}+\lambda(Ax^{k+1}+Bv^{k+1}-c).\end{aligned} \tag{2.15}$$

Steps one and two update x and v respectfully, in an alternating fashion. Step three is the dual variable update. By introducing the scaled dual variable $u=\frac{1}{\lambda}y$, the ADMM algorithm becomes simpler and more intuitive for our purposes. The new form is called

the scaled form, and the augmented Lagrangian can be rewritten as:

$$L_\lambda = f(x) + g(v) + \frac{\lambda}{2}\|Ax + Bv - c + u\|_2^2 - \frac{\lambda}{2}\|u\|_2^2, \tag{2.16}$$

The scaled ADMM is shown in Algorithm (2.1). This version is often more convenient to use. In particular, regarding our work, some observations are seen more clearly in this formulation, and we thus later refer only to it.

Algorithm 2.1 Scaled ADMM algorithm

Initialization: set $k = 0$, set u^0 and v^0 to some initialization;
while !stopping criteria **do**
$x^{k+1} = \arg\min_x f(x) + \frac{\lambda}{2}\left\|Ax + Bv^k - c + u^k\right\|_2^2$
$v^{k+1} = \arg\min_v g(z) + \frac{\lambda}{2}\left\|Ax^{k+1} + Bv - c + u^k\right\|_2^2$
$u^{k+1} = u^k + Ax^{k+1} + Bv^{k+1} - c$
$k = k + 1$
end while
Output: x^k

Under some reasonable conditions (i.e f and g are closed, proper and convex, and the unaugmented Lagrangian L_0 has a saddle point – Refer [BPC+11] for additional information) ADMM is guaranteed to converge to the proper and unique solution. Under additional conditions ADMM is guaranteed to converge even when each of the above steps is computed inexactly [BPC+11], [SVW+15]. This is useful, for instance, when v^{k+1} is given by an expression that only approximates the actual minimization result.

The convergence rate of ADMM is quite slow and it requires many iterations to get to the exact solution. However, for practical scenarios, a modestly accurate solution is sufficient, and ADMM often achieves this quite fast (e.g [ABDF10]). The choice of the parameter λ has a great impact on the convergence speed in practical scenarios, and thus should be chosen wisely. ADMM has drawn a lot of interest in recent years. More information about ADMM and its convergence analysis can be found in [BPC+11].

Chapter 3

Previous Work

Restoration of Poisson noisy images was studied in various papers. There are two main approaches. The first is to transform the problem into a Gaussian restoration one (e.g. [FN04], [ZFS08]). The second is to perform the restoration process directly on the Poisson noisy image (e.g. [SHDW14], [GE14]). In this chapter we discuss both approaches.

3.1 Variance Stabilizing Transform Based Methods

If we apply a function $f(\cdot)$ on a random variable, its probability density function will change. It is possible to use this fact to approximately transform an image contaminated with Poisson noise into one with additive white Gaussian noise. This is helpful because there exists a vast amount of well studied, powerful restoration algorithms for problems with Gaussian noise. The function $f(\cdot)$ should be such that the variance of the transformed variable is not dependent on its mean. Such functions are called Variance Stabilizing Transformations (VST).

For our purposes it is of interest to find such a transform that approximately turns a Poisson random variable into a Gaussian one that has roughly a unit variance. One example for such a transformation is the function $f(y) = 2\sqrt{y}$, where y is our Poisson distributed random variable. Anscombe [Ans48] showed that it is usually better to use $f(y) = 2\sqrt{y + \frac{3}{8}}$, as this function provides a better variance stabilization on the peak ranges that are relevant to images. Using this transform we can construct the naïve Algorithm 3.1 that applies a Gaussian restoration (denoted by $\mathfrak{R}(\cdot)$) preceded by the Anscombe transform and followed by its algebraic inverse.

Algorithm 3.1 Poisson restorotaion based on Anscombe transform with naïve inverse transform

Input: Degraded image y, Gaussian restoration algorithm $\mathfrak{R}(\cdot)$;
Step 1: $y_{Ansc} = 2\sqrt{y + \frac{3}{8}}$
Step 2: $\hat{x}_{Ansc} = \mathfrak{R}(y_{Ansc})$
Step 3: $\hat{x} = (\frac{1}{2}\hat{x}_{Ansc})^2 - \frac{3}{8}$
Output: $\hat{x}$

The third step of Algorithm 3.1 which is applying the algebraic inverse of the Anscombe function in order to remove the distortion created by this transformation at the first step, is actually inaccurate. This is due to the fact that the Anscombe transform is non-linear and thus

$$E\big[f(x)|\mu\big] = \sum P(x|\mu)f(x) \neq f\Big(\sum P(x|\mu)x\Big) = f\big(E\left[x|\mu\right]\big), \tag{3.1}$$

where μ is the mean of x, which is equal to the clean pixel value. This leads to the following inequality:

$$f^{-1}\Big(E\big[f(x)|\mu\big]\Big) \neq E\left[x|\mu\right]. \tag{3.2}$$

This implies that the naïve algebraic inverse transform is biased, and in order to get better reconstruction a different one should be used, which does not violate the above. To have an unbiased inverse we need to look at a transform T that obeys the relation:

$$T\Big(E\big[f(x)|\mu\big]\Big) = \mu. \tag{3.3}$$

By substituting f with the Anscombe transform and using the definition of the mean we have

$$E\left[f\left(x|\mu\right)\right] = \sum_{k=0}^{\infty} 2\sqrt{x + \frac{3}{8}} \cdot \frac{\mu^x e^{-\mu}}{x!}. \tag{3.4}$$

Thus, T must obey:

$$T\left(\sum_{k=0}^{\infty} 2\sqrt{x + \frac{3}{8}} \cdot \frac{\mu^x e^{-\mu}}{x!}\right) = \mu. \tag{3.5}$$

By plotting μ as a function of $\sum\limits_{k=0}^{\infty} 2\sqrt{x + \frac{3}{8}} \cdot \frac{\mu^x e^{-\mu}}{x!}$ for enough values, we can create a look-up table and calculate the inverse transformation at any given point by using a simple interpolation. This process is done only once, and offline. This transform can replace step 3 in Algorithm 3.1 to produce superior results. This improved inverse transformation was proposed in [MF11]. Obviously, this method to construct an unbiased inverse transform could be applied to other VSTs.

The Anscombe transform works well only for large μ values. In this case the transformed random variable is indeed distributed normally with $\sigma = 1$. However, when

using this transform on a Poisson random variable with a small value of μ, it behaves differently, and cannot be restored accurately by Gaussian restoration algorithms. Figure 3.1 shows the variance of the random variable that is produced by the Anscombe transform as a function of its mean μ. The result clearly show that for $\mu < 4$ this transform is ineffective.

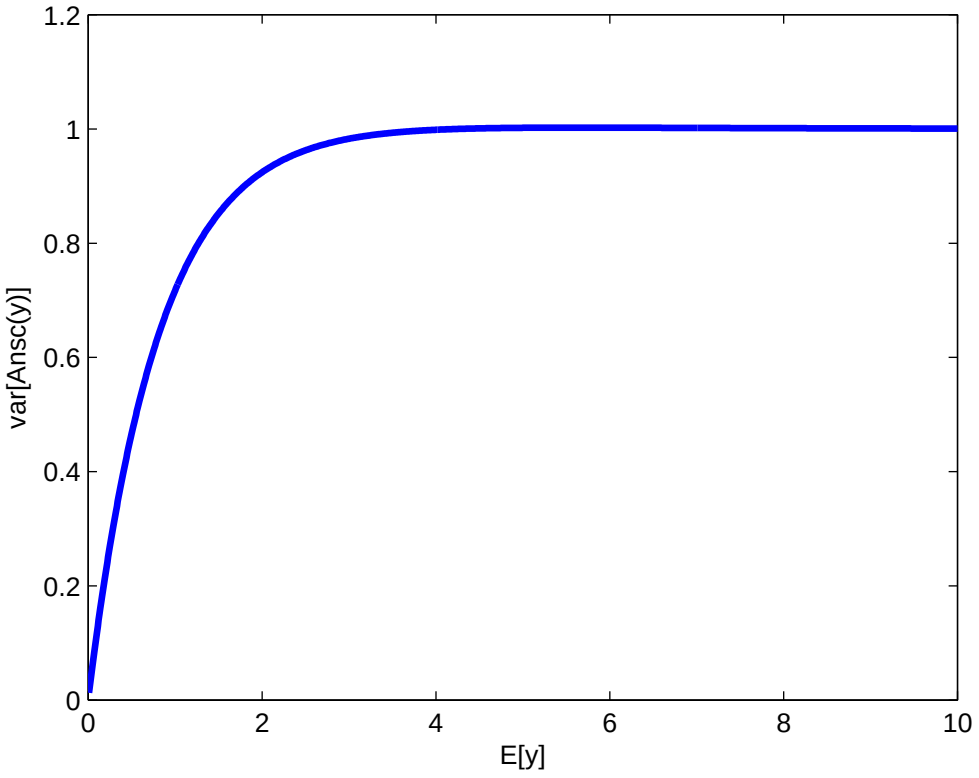


Figure 3.1: σ^2 of Anscombe tranform as a function of μ

Clearly, for small mean values, μ, the VST proposed is ineffective. Therefore there is a need for another approach which treats the Poisson noise directly and is effective for small peak values as well as for large ones.

3.2 Direct Reconstruction Method

The second approach to restore Poisson noisy images is to directly treat the Poisson distribution. Methods that take this path usually perform better than VST-based algorithms, especially in the lower peak regime.

We start by recalling Proposition 2.2.2, which states that the MAP estimator of a Poisson noisy image is

$$\hat{x} = \arg\min_{x} -y^T ln(x) + 1^T x - \ln\left(P\left(x\right)\right). \tag{3.6}$$

The regularization term $-ln(P(x))$ incorporates the prior knowledge we have on the clean image. If we simply choose to neglect this term, we actually assume that each x has the same probability to represent an actual image. The known Maximum Likelihood Estimator (MLE) is the solution to this problem. When considering the denoising case we can get the following analytical solution by simply differentiating the objective function in Equation (3.6) and zeroing it:

$$\nabla_x(-y^T ln(x) + 1^T x) = -y./x + 1 = 0, \tag{3.7}$$

where ./ stands for an element-wise division and 1 is a vector of all ones. This gives the analytical solution $x = y$, which means that the "denoised" reconstruction is the noisy image itself. Obviously this is useless, and thus the prior term is needed.

Several effective regularizations were suggested in the literature, which led to good reconstruction results. Those regularizations were typically used in a Gaussian framework, but as the prior knowledge does not depend on the inversion problem, they can be used in other problems, such as ones with Poisson noise. We now survey several recent priors and their corresponding regularization terms. We present techniques from the Gaussian framework, and explain how they may be generalized to other cases as well.

3.2.1 The Self-Similarity Model

Self-similarities are common in natural images. This property has led to a number of powerful restoration algorithms, all sharing the same concept: For each patch, find a group of similar patches. Then, reconstruct each of them by a weighted average of all other similar corresponding patches. The most popular algorithm that takes this approach is the Non-Local Means (NLM) scheme [BCM05]. This approach was also applied in [DTD10] for Poisson denoising. Another strategy that uses self similarities in images was introduced in the block-matching and 3D filtering (BM3D) method [DFKE07]. Here, similar patches are stacked in a 3D tensor, and a hard thresholding is applied on their coefficients in the 3D-DCT domain. Initial version of the reconstructed image is achieved by putting the patches back into their original locations. The final recovery is generated by repeating the same process but with a Wiener filter, which relies on the initial reconstruction of the first phase of the algorithm, instead of the hard thresholding step.

3.2.2 The Sparsity Model

The Sparsity model [Ela10] builds upon the assumption that natural signals are generated out of a small amount of basis elements in some space. Algorithms relying on this model have been shown to lead to very good results. The basic optimization problem (for the

Gaussian framework) that corresponds to this model is

$$\begin{aligned} &\arg\min_{\alpha} \ \|\alpha\|_0 \\ &s.t \ \ \|y - D\alpha\|_2^2 \leq \sigma^2, \end{aligned} \tag{3.8}$$

where the reconstructed image is $\hat{x} = D\alpha$. The matrix D, which is not necessarily a square matrix (in the case where the number of columns in D is larger than the number of rows, D is referred to as an over-complete dictionary), is called the dictionary. The vector α is called the sparse code and it is assumed to have a small amount of non zeros, a property which is enforced by the L_0 pseudo-norm. As L_0 is non convex, it is common to replace it with the L_1 norm , which is the "closest" convex norm. It has been shown that this norm also promotes sparsity in many cases [DH01]. Rewriting Equation (3.8) in an unconstrained form with the L_1 norm instead of the L_0 one, provides us with

$$\hat{x} = \arg\min_{\alpha} \|y - D\alpha\|_2^2 + \beta\|\alpha\|_1, \tag{3.9}$$

where β should be chosen such the constraint $\|y - D\alpha\|^2 \leq \sigma^2$ is met. Note that this term is of the same form as the MAP for Gaussian noise with $\|\alpha\|_1$ standing as the prior regularization term.

Many enhancements to the model were suggested in the literature, often leading to superior results [MBP+09],[EA06]. A common approach is to find sparse codes for every patch and estimate it, instead of the entire image. The final image can be reconstructed by averaging overlapping patches. Another improvement considers the selection of the correct dictionary D. The dictionary D can be pre-defined such as the DCT, or it can be learned [EA06]. Learned dictionaries have been demonstrated to produce superior results.

An additional gain to the scheme may be achieved by using simultaneous sparse coding [MBP+09]. In this scheme similar patches are encouraged to have a similar sparse coding support (non zero locations). This strategy enjoys the advantages of both the sparsity model and the self similarity model mentioned earlier.

We now turn to describe two algorithm which suggest specific reconstruction algorithm of Poisson noisy images using the above ideas.

3.3 Direct Poisson Denoising Algorithms

3.3.1 The Non-Local PCA (NLPCA) and variations

By using the sparsity model prior, together with a dictionary learning technique, a possible strategy for denoising a Poisson noisy signal y is

$$\arg\min_{\alpha} -y^T ln(D\alpha) + 1^T D\alpha + \lambda\|\alpha\|_0. \tag{3.10}$$

If we are to treat a large image, it can be broken into all its overlapping patches, such that we sum the above log-posterior over all the patches, in an attempt to find the global denoised result. This leads to the following MAP approach:

$$\underset{\alpha_1,\alpha_2,\ldots,\alpha_N}{\arg\min} \sum_{i=1}^{N} -y^T ln(D\alpha_i) + 1^T D\alpha_i + \lambda\|\alpha_i\|_0 \tag{3.11}$$

This optimization problem can be also used to find the dictionary D that best fits the given image y. This implies that the minimization is done w.r.t. to D as well, in the spirit of the image denoising algorithm introduced in [EA06].

The problem with the above scheme is the fact that $D\alpha_i$ appears inside the $ln(\cdot)$, which requires us to add constraints on the positivity of $D\alpha_i$. Such constraints will necessarily complicate the overall scheme. As a remedy to this problem, in the non-local PCA (NLPCA) algorithm, Salmon et. al. [SHDW14] proposed to define the local recovered patches as $e^{D\alpha_i}$. Thus we would like to solve

$$\underset{D,\alpha_1,\alpha_2,\ldots,\alpha_N}{\arg\min} \sum_{i=1}^{N} -y^T D\alpha + 1^T e^{D\alpha} + \lambda\|\alpha\|_0. \tag{3.12}$$

Here the authors suggest to partition the image into overlapping patches, group the patches into disjoint groups and solve for each group the minimizing problem

$$\underset{D,\alpha_1,\alpha_2,\ldots,\alpha_N}{\arg\min} \sum_{i=1}^{N} 1^T e^{D\alpha_i} - y^T D\alpha_i \tag{3.13}$$

where D is a dictionary with only k columns, associated to the group, and α_i is the sparse codes of patch i. Because D has only k columns, the sparse codes vectors α_i are k-sparse by the construction. Finally, the i-th patch gets the value $e^{D\alpha_i}$ and is returned to its location. The further improved non-local sparse PCA (NLSPCA) algorithm [SHDW14] adds a L_1 penalty term and minimizes for each group the expression

$$\underset{D,\alpha_1,\alpha_2,\ldots,\alpha_N}{\arg\min} \sum_{i=1}^{N} 1^T e^{D\alpha_i} - y^T D\alpha_i + \lambda \sum_{i=1}^{N} \|\alpha_i\|_1, \tag{3.14}$$

In both algorithms, each group has its own narrow D. This may weaken the expressibility of this model, resulting in weaker denoising effect.

3.3.2 The Sparse Poisson Denoising algorithm (SPDA)

An improvement to the previous method is suggested in the sparse Poisson denoising algorithm (SPDA) [GE14]. As in the NLPCA and NLSPCA algorithms, here too, the patches are arranged in disjointed groups. However, in SPDA, the sizes of the groups are much smaller and a global dictionary is learned for all sets of patches together, instead of learning a unique dictionary for each group. The minimization objective of

SPDA can be formulated as

$$\begin{aligned} &\arg\min_{D,\alpha_1,\alpha_2,\ldots,\alpha_N} \sum_{i=1}^{N} 1^T e^{D\alpha_i} - y^T D\alpha_i. \\ &s.t \quad \|\alpha_i\|_0 \leq k \end{aligned} \tag{3.15}$$

In SPDA a simultaneous sparse coding is used for each group of patches. This generates a global force on the restoration process, which produces superior denosing results.

Chapter 4

The Plug-and-Play Priors (PaPP) Approach

4.1 Standard Plug-and-Play Prior

The Plug-and-Play Prior (PaPP) framework, proposed by Venkatakrishnan, Bouman and Wohlberg [VBW13], allows simple integration between inverse problems and priors, by applying a Gaussian denoising algorithm, which corresponds to the used prior. One of the prime benefits in the PaPP 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 ADMM optimization method [BPC$^+$11], Then the prior is deployed indirectly by activating a Gaussian denoising algorithm of choice. For specific such algorithms, the convergence is proven and guaranteed [SVW$^+$15].

The goal of the PaPP framework is to maximize the posterior probability in an attempt to implement the MAP estimator. Mathematically, this translates to the following:

$$\max_{x \in R^{m \times n}} P\left(x|y\right) = \max_{x \in R^{m \times n}} \frac{P\left(y|x\right) P\left(x\right)}{P\left(y\right)} = \max_{x \in R^{m \times n}} P\left(y|x\right) P\left(x\right). \tag{4.1}$$

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 $-ln\left(\cdot\right)$ of this expression gives an equivalent problem of the form

$$\min_{x \in R^{m \times n}} -ln\left(P\left(x|y\right)\right) = \min_{x \in R^{m \times n}} -ln\left(P\left(y|x\right)\right) - ln\left(P\left(x\right)\right). \tag{4.2}$$

In order to be consistent with the notations in [VBW13], we denote $l\left(x\right) = -ln\left(P\left(y|x\right)\right)$ and $-ln(P(x)) = \beta s(x)$, where $s(x)$ is a prior on x associated with a denoising algorithm $\mathfrak{D}_\sigma(\cdot)$ such that

$$\mathfrak{D}_\sigma(y) = \arg\min_{x \in R^{m \times n}} \frac{1}{2}||y - x||_2^2 + \sigma^2 s(x). \tag{4.3}$$

Note that the constant β sets the weight of the prior in the optimization problem and is dependent on the noise variance. Thus our task is to find x that solves the problem

$$\hat{x} = \underset{x \in R^{m \times n}}{\arg\min}\, l(x) + \beta s(x). \tag{4.4}$$

Note that in both Equations (4.2) and (4.3) y is constant. By using a variable splitting technique in the optimization problem, we get

$$\begin{aligned} \hat{x} = & \underset{x,v \in R^{m \times n}}{\arg\min}\; l(x) + \beta s(v). \\ & s.t\;\; x = v \end{aligned} \tag{4.5}$$

This problem can be solved using ADMM [BPC+11] by constructing an augmented Lagrangian which is given by

$$L_\lambda = l(x) + \beta s(v) + \frac{\lambda}{2} \|x - v + u\|_2^2 - \frac{\lambda}{2} \|u\|_2^2. \tag{4.6}$$

ADMM theory [BPC+11] states that (under proper conditions) minimizing Equation (4.5) is equivalent to iterating until convergence over the following three steps:

$$\begin{aligned} x^{k+1} &= \underset{x}{\arg\min}\, L_\lambda\left(x, v^k, u^k\right), \\ v^{k+1} &= \underset{v}{\arg\min}\, L_\lambda\left(x^{k+1}, v, u^k\right), \\ u^{k+1} &= u^k + \left(x^{k+1} - v^{k+1}\right). \end{aligned} \tag{4.7}$$

By plugging L_λ we get

$$\begin{aligned} x^{k+1} &= \underset{x}{\arg\min}\, l(x) + \tfrac{\lambda}{2} \left\|x - \left(v^k - u^k\right)\right\|_2^2, \\ v^{k+1} &= \underset{v}{\arg\min}\, \tfrac{\lambda}{2} \left\|x^{k+1} + u^k - v\right\|_2^2 + \beta s(v), \\ u^{k+1} &= u^k + \left(x^{k+1} - v^{k+1}\right). \end{aligned} \tag{4.8}$$

There exist ADMM variations that modify λ at each iteration. Thus λ could be dependent on k. For several simple cases λ can be set analytically. One such scenario is shown in Appendix A. The first step is dependent on the inversion problem we are trying to solve. Note that the second step corresponds to applying the Gaussian denoising algorithm $\mathfrak{D}_{\beta/\lambda}$, on the image $x^{k+1} + u^k$ with variance of $\sigma^2 = \frac{\beta}{\lambda}$. Therefore, as already mentioned above, we do not have to know the formulation of the prior explicitly, as we can simply use a Gaussian denoising algorithm which corresponds to it. The third step is a dual variable update. The PaPP framework's convergence was analyzed in [SVW+15]. 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.

In Chapter 5 we show how PaPP may be used for Poisson inverse problems. We

shall see that in this case, step 1 in Equation (4.8) is convex and can be easily computed. When handling the Poisson denoising problem, this step turns to be even simpler as it is also separable, resulting with a scalar formula that resembles the Anscombe transform.

4.2 Plug-and-Play Priors Extention

4.2.1 Plug-and-Play with Multiple Priors

We can propose a simple extension of the Plug-and-Play Priors method that enables the usage of several Gaussian denoisers. We start from the following ADMM formulation, which follows Equation (4.5)

$$\begin{aligned} &\arg\min\ l(x) + \beta s(v) \\ &s.t\ Ax - Bv = c \end{aligned} \tag{4.9}$$

By choosing

$$\begin{aligned} &A = \begin{bmatrix} I \\ I \end{bmatrix} \Rightarrow Ax = \begin{bmatrix} x \\ x \end{bmatrix}, \\ &v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}, B = I \Rightarrow Bv = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}. \\ &c = 0, \end{aligned} \tag{4.10}$$

and if $\beta s(v) = \beta_1 s_1(v_1) + \beta_2 s_2(v_2)$ we get the equivalent formulation

$$\begin{aligned} &\arg\min\ l(x) + \beta_1 s_1(v_1) + \beta_2 s_2(v_2), \\ &s.t\ x = v_1, x = v_2 \end{aligned} \tag{4.11}$$

where s_1 and s_2 are two priors that we aim to use, and v_1 and v_2 are two auxiliary variables that will help in simplifying the solution of this problem. Following the steps taken above in the derivation of the PaPP, we have

Step 1:

$$x^{k+1} = \arg\min_x l\left(x\right) + \lambda \left\| x - {v_1}^k + {u_1}^k \right\|_2^2 + \lambda \left\| x - {v_2}^k + {u_2}^k \right\|_2^2, \tag{4.12}$$

As in the regular PaPP, this expression is convex if $l(x)$ is convex, and separable if $l(x)$ is separable.

Step 2:

$$\begin{aligned} {v_1}^k &= \arg\min_v \beta_1 s_1\left(v_1\right) + \lambda \left\| x^{k+1} - v_1 + {u_1}^k \right\|_2^2, \\ {v_2}^k &= \arg\min_v \beta_2 s_2\left(v_2\right) + \lambda \left\| x^{k+1} - v_2 + {u_2}^k \right\|_2^2, \end{aligned} \tag{4.13}$$

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

Step 3:

$$\begin{aligned} {u_1}^{k+1} &= u_1^k + x^{k+1} - {v_1}^{k+1}, \\ {u_2}^{k+1} &= u_2^k + x^{k+1} - {v_2}^{k+1}, \end{aligned} \tag{4.14}$$

this step updates the dual variables.

Obviously, this scheme can be generalized to use as many priors as needed. The core idea behind this generalization is that often times we may encounter different priors that address different features of the unknown image, such as self-similarity; local smoothness or other structure; scale-invariance; etc. Merging of two priors into the PaPP scheme may lead to an overall benefit, 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 $s_2(v)$.

The idea of using several prior terms has appeared previously in the context of Poisson noise in [PCP11]. 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.

4.2.2 Over Relaxed Plug-and-Play Priors

The over relaxed ADMM states that it is possible to replace the x^{k+1} term with $\alpha x^{k+1} - (1-\alpha)z^k$ in the v^{k+1} and u^{k+1} update steps, where $\alpha \in (0, 2]$. When $\alpha > 1$ this version of ADMM is called over relaxed. This version can, in many scenarios, results with faster convergence.

The resulting PaPP formulation that corresponds to the over relaxed ADMM is

$$\begin{aligned} x^{k+1} &= \arg\min_x l(x) + \frac{\lambda}{2}\left\|x - v^k + u^k\right\|_2^2, \\ \tilde{x}^{k+1} &= \alpha x^{k+1} - (1-\alpha)z^k, \\ v^{k+1} &= \arg\min_v \frac{\lambda}{2}\left\|\tilde{x}^{k+1} + u^k - v\right\|_2^2 + \beta s(z), \\ u^{k+1} &= u^k + \tilde{x}^{k+1} - v^{k+1}. \end{aligned} \tag{4.15}$$

The x update step is convex if $f(\cdot)$. The v update step, just like in the regular PaPP, means applying a Gaussian denoising algorithm.

Chapter 5

Plug and Play Priors for Poisson Inverse Problems

5.1 The core P^{4}IP Algorithm

We now turn to introduce the "Plug-and-Play Prior for Poisson Inverse Problem" algorithm (P^{4}IP), and how it uses the plug-and-play Priors framework. We start by invoking the proper log-likelihood function $l(x)$ into the above-described formulation, enabling the integration of Gaussian denoising algorithms to the Poisson inverse problems. Then we discuss two applications of our algorithm – the denoising and the deblurring scenarios.

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's 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 we should find an expression for $l(x)$. As mentioned before, this is given by taking $-ln(\cdot)$ of $P(y|x)$. When taking H into account we have

$$P(y|x) = \prod_i \frac{(Hx)[i]^{y[i]}}{\Gamma(y[i]+1)} e^{-(Hx)[i]}. \tag{5.1}$$

Thus, $l(x)$ is given by

$$\begin{aligned} l(x) = -\ln(P(y|x)) &= -\sum_i \ln\left(\frac{(Hx)[i]^{y[i]}}{\Gamma(y[i]+1)} e^{-(Hx)[i]}\right) \\ &= -y^T \ln(Hx) + 1^T Hx + constant. \end{aligned} \tag{5.2}$$

Relying on Equation (4.8), the first ADMM step in matrix form is therefore

$$\arg\min_x L_\lambda = \arg\min_x -y^T \ln(Hx) + 1^T Hx + \frac{\lambda}{2} \|x - v + u\|_2^2. \tag{5.3}$$

This expression is convex and can be solved quite efficiently by modern optimization methods. By using Equation (5.3), we get Algorithm 5.1. The parameter λ can be changed in many forms in each iteration. We chose to use the update rule $\lambda^k = \lambda_0 \cdot (\lambda_{step})^k$, where λ_0 and λ_{step} are some constants.

Algorithm 5.1 - P^{4}IP

Input: Distorted image y, Gaussian denoising algorithm $\mathfrak{D}_\sigma(\cdot)$;
Initialization: set $k = 0$,$u^0 = 0, v^0$ =some initialization, $\lambda_0 = const$;
while !stopping criteria **do**
$x^{k+1} = \arg\min_x -y^T \ln(Hx) + 1^T Hx + \frac{\lambda^k}{2} \left\| x - v^k + u^k \right\|_2^2$
$v^{k+1} = \mathfrak{D}_\sigma(x^{k+1} + u^k)$ with $\sigma^2 = \frac{\beta}{\lambda^k}$
$u^{k+1} = u^k + \left(x^{k+1} - v^{k+1}\right)$
$\lambda^{k+1} = \lambda^k \cdot \lambda_{step}$
$k = k + 1$
end while
Output: Reconstructed image x^k

5.2 M-P^{4}IP Algorithm

Obviously we could use the Plug-and-Play Priors extension that employs several denoising methods, as shown in the previous section. Such a change requires only slight modifications to Algorithm 5.1. The resulted M-P^{4}IP algorithm is shown in 5.2.

Algorithm 5.2 M-P^{4}IP

Input: Distorted image y, Gaussian denoising algorithms $\mathfrak{D}_\sigma^1(\cdot)$ and $\mathfrak{D}_\sigma^2(\cdot)$
Initialization: set $k = 0$, $u_1^0 = 0, v_1^0$ =some initialization,
$u_2^0 = 0, v_2^0$ =some initialization, $\lambda_0 = const$;
while !stopping criteria **do**
$x^{k+1} = \arg\min_x l(x) + \lambda^k \left\| x - {v_1}^k + {u_1}^k \right\|_2^2 + \lambda^k \left\| x - {v_2}^k + {u_2}^k \right\|_2^2$.
$v_1^{k+1} = \mathfrak{D}_\sigma^1(x^{k+1} + u_1^k)$ with $\sigma^2 = \frac{\beta_1}{\lambda^k}$
$v_2^{k+1} = \mathfrak{D}_\sigma^2(x^{k+1} + u_2^k)$ with $\sigma^2 = \frac{\beta_2}{\lambda^k}$
${u_1}^{k+1} = u_1^k + x^{k+1} - {v_1}^{k+1}$
${u_2}^{k+1} = u_2^k + x^{k+1} - {v_2}^{k+1}$
$\lambda^{k+1} = \lambda^k \cdot \lambda_{step}$
$k = k + 1$
end while
Output: Reconstructed image x^k

5.3 Poisson Denoising

For the special case of Poisson denoising $H = I$. The solution to the first ADMM step can be solved by a closed form solution, as can be seen in the following theorem.

Theorem 5.1. *When $H = I$ the first ADMM step is separable, and can be solved for each pixel individually. Moreover, this step can be solved by the closed form solution*

$$x^{k+1}[i] = \frac{\left(\lambda\left(v^k[i] - u^k[i]\right) - 1\right) + \sqrt{\left(\lambda\left(v^k[i] - u^k[i]\right) - 1\right)^2 + 4\lambda y[i]}}{2\lambda}, \tag{5.4}$$

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).

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

$$l\left(X\right) = -y^T \ln\left(x\right) + 1^T x. \tag{5.5}$$

The augmented Lagrangian is thus,

$$L_\lambda = -y^T \ln\left(x\right) + 1^T x - \beta \ln\left(P\left(v\right)\right) + \frac{\lambda}{2}\left\|x - v + u\right\| - \frac{\lambda}{2}\left\|u\right\|_2^2, \tag{5.6}$$

and the first ADMM step becomes

$$x^{k+1} = \arg\min_x L_\lambda\left(x, v^k, u^k\right) = \arg\min_x -y^T \ln\left(x\right) + 1^T x + \frac{\lambda}{2}\left\|x - v^k + u^k\right\|_2^2. \tag{5.7}$$

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 $x\left[i\right]$ and equating to 0 we get

$$-\frac{y[i]}{x[i]} + 1 + \lambda\left(x[i] - v^k[i] + u^k[i]\right) = 0. \tag{5.8}$$

Thus, we have

$$x[i] = \frac{\left(\lambda\left(v^k[i] - u^k[i]\right) - 1\right) + \sqrt{\left(\lambda\left(v^k[i] - u^k[i]\right) - 1\right)^2 + 4\lambda y[i]}}{2\lambda}. \tag{5.9}$$

As y is non negative, the expression inside the square root is also non negative and causes the resulted x to be non negative as well. Another possible solution could have been the second root of Equation (5.8), but this solution is purely negative and thus uninformative. □

Such a result makes each P^{4}IP iteration as efficient as the Gaussian denoising algorithm in use.

A closer look at the expression given in Equation (5.4) 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 (5.4) is applied onto it is given by $2\sqrt{\frac{3}{8}} + 2\sqrt{\frac{3}{8} + y}$. Thus the variance is the same as the one achieved by Anscombe's transform, because the two differ only by a constant. Figure 5.1 shows

the Anscombe transform and the one obtained by Equation (5.4) 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 (5.4) may look like the Anscombe one, P4IP is substantially different than the Anscombe transform based denoising in two ways: (i) The curve by Equation (5.4) changes (locally) from one iteration to another due to the change in u and v, and; (ii) P4IP has no inverse transform step after the Gaussian denoising. This leads to improved restoration compared to the classical VST denoising scheme.

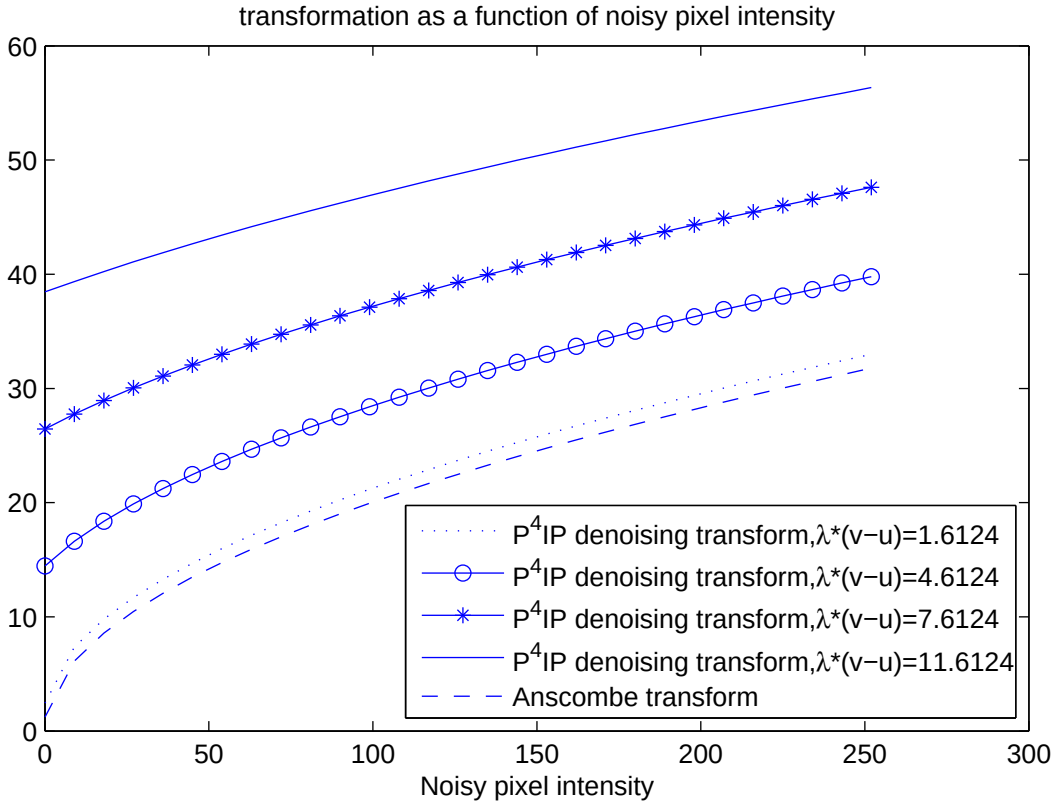


Figure 5.1: P^{4}IP and Anscombe transformations as a function of the current intensity of the noisy images. $\lambda = 0.25$.

5.4 Poisson Deblurring

When dealing with the deblurring problem, H represents a blur matrix. The first PaPP 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

$$\nabla_x L_\lambda = -H^T\left(y./\left(Hx\right)\right) + H^T 1 + \lambda\left(x - v + u\right). \tag{5.10}$$

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.

5.5 Relation to previous work

In the context of Gaussian deblurring, a similar derivation is adapted in [DKE10]. Here too, the authors use a splitting of the fidelity term and a prior term, but unlike the PaPP, an explicit formulation of the prior is required. The authors chose to use a spatially adaptive filter, which groups similar patches, applies the thresholding operator in a 3-D transform which consists of a 2-D discrete sine and 1-D Haar transforms, and finally restoring the patches to their original locations.

This splitting of the two terms appears also in Poisson restoration algorithms [DFS12],[FBD10], and is shown to be a beneficial technique in this scenario as well. The work reported in [DFS12] relies on the proximal splitting tool, 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 [FBD10] uses the ADMM technique to enable the splitting. The authors test the Total Variation (TV) and analysis and synthesis priors. In [MMYZ13] 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 [PCP11]. 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.

Chapter 6

P^{4}IP Experiments

6.1 Implementation Details

In both the denoising and deblurring scenarios, we use BM3D as our Gaussian denoiser, as it provides very good results in the Gaussian case and has an efficient implementation. In the denoising scenario we also test the multiple prior algorithm, M-P^{4}IP. As our second denoiser we use a simplified version of the multi-scale KSVD algorithm [SOE14], which takes into account multi-scale considerations in the denoising process. These 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 in the denoising problem, an improvement in the recovery can be achieved by using a technique called binning: The noisy image is first down-scaled by aggregating neighbor pixels, which leads to a higher photon count in each pixel in the resulted smaller image and therefore also a better SNR, and then the denoising algorithm is applied on this smaller sized image. Mathematically, the binned image is $y_{bin} = SBy$ where B is the uniform blur operator and S is the sampling operator. An estimate for the high resolution image is obtained by applying a simple linear interpolation on the denoised small image. In addition to the better recovery quality, the binning strategy also reduces the runtime as the denoising is applied to smaller images. In our experiments we use a 3:1 down-scaling in each axis for the binning. Binning can only be used in the denoising scenario as down-sampling does not commute with the blur operator. This can be seen by looking at the binned version of an image blurred by some kernel H:

$$SBHy \neq HSBy. \tag{6.1}$$

For a general blur matrix H, equality does not necessarily hold. This means that 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 up-scaling stage, where artifacts from the restoration algorithm are amplified during this process. Thus

using binning in the debluring 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 in Section 5.4, when dealing with the deblurring scenario, P^{4}IP requires an optimization technique that can find a minimum of the first PaPP step:

$$x^{k+1} = \arg\min_x -y^T \ln(Hx) + 1^T Hx + \frac{\lambda}{2}\left\|x - v^k + u^k\right\|_2^2. \tag{6.2}$$

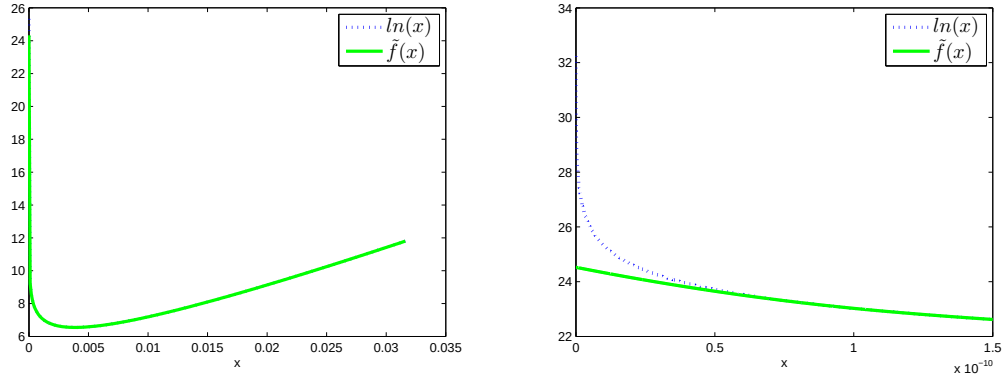
L-BFGS was empirically chosen for this task. In order to avoid calculating $\ln(\cdot)$ on negative Hx values, we modify the $ln(\cdot)$ term by the surrogate function

$$\tilde{f}(z) = \begin{cases} ln(z) \ , & z \geq \varepsilon \\ az^2 + bz + c \ , & z < \varepsilon. \end{cases} \tag{6.3}$$

We thus approximate the first PaPP step with:

$$x^{k+1} = \arg\min_x -y^T \tilde{f}(Hx) + 1^T Hx + \frac{\lambda}{2}\left\|x - v^k + u^k\right\|_2^2 \tag{6.4}$$

instead of Equation (6.2). The coefficients a, b and c are chosen such that $\tilde{f}(\cdot)$ and $\nabla\tilde{f}(\cdot)$ coincides with $ln(\cdot)$ and its derivative at $x = \epsilon$. Thus we have that $a = \frac{-1}{2\epsilon^2}, b = \frac{2}{\epsilon}$ and $c = ln(\epsilon) - 1.5$. Obviously this is not the only option for choosing those coefficients. As $x \to 0$ we get that $Hx \to 0$ and $-ln(Hx) \to \inf$. Therefore choosing a small enough value for ϵ, guarantees that by using the surrogate function in Equation (6.4) the same minimum is produces as in (6.2) and that all entries in Hx that $ln(\cdot)$ is applied on, are positive. We chose to use $\epsilon = 10^{-10}$. Figure 6.1 shows visually the difference between both objective functions for a simple case where H is constant. For this case, it is obvious that both objective have the same minimum, and thus using $\tilde{f}(\cdot)$ instead of $ln(\cdot)$ doesn't effect the result of Equation (6.2). However, $ln(\cdot)$ is not applied on negative values.



(a) Full scale (b) Zoom on scale near ϵ

Figure 6.1: Objective functions comparison. The minimum of both function is equal.

6.1.1 Choice of Parameters

Both denoising and deblurring scenarios require the appropriate choice of parameters. 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. Figure 6.2 shows the average PSNR on eight test images as a function of β near β's optimal value.

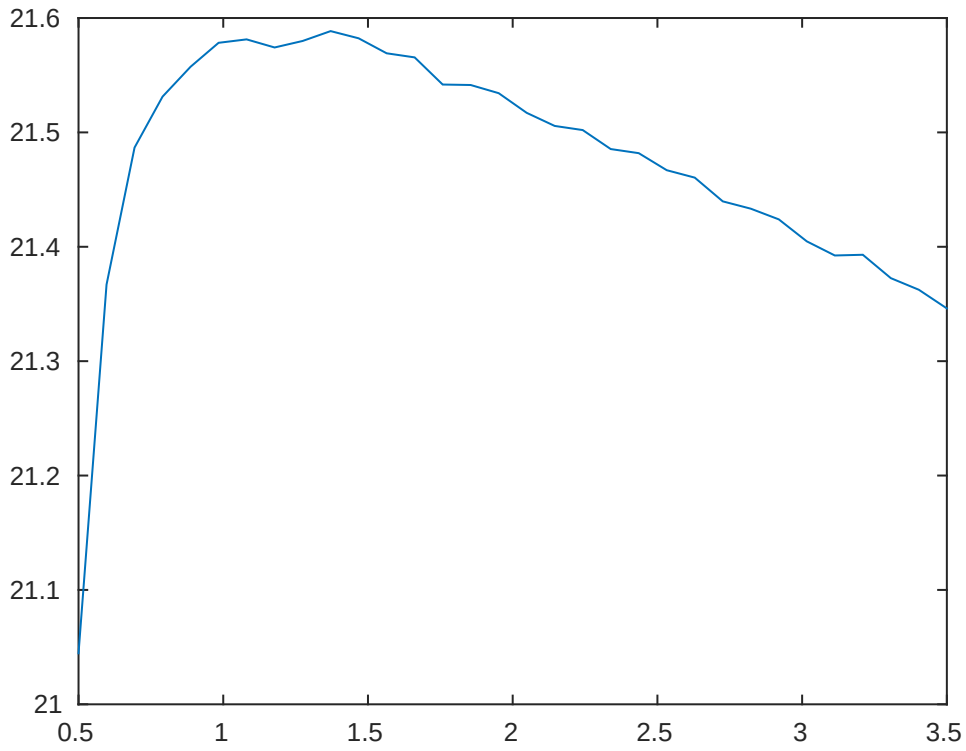


Figure 6.2: average PSNR on all test images as a function of β

Interestingly, the optimal β can be approximated by the simple polynomial $peak^{-0.75}$. Figure 6.3 show the optimal β and the polynomial approximation of it as a function of the peak. The optimal β was found by an exhaustive search.

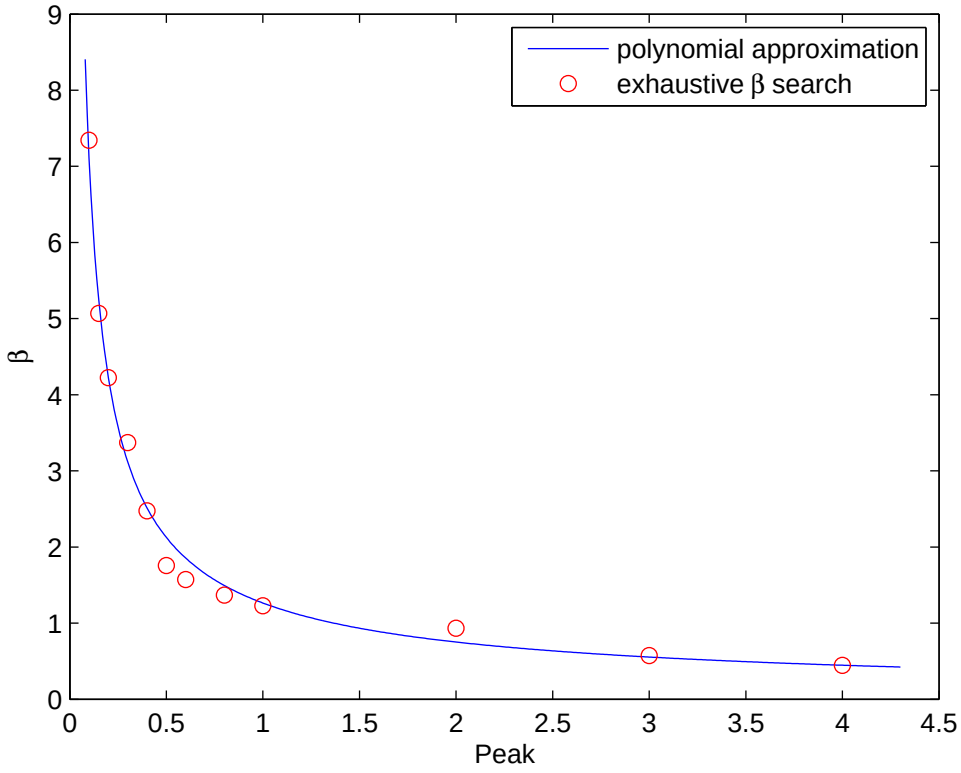


Figure 6.3: Optimal β and polynomially approximated β as a function of the peak.

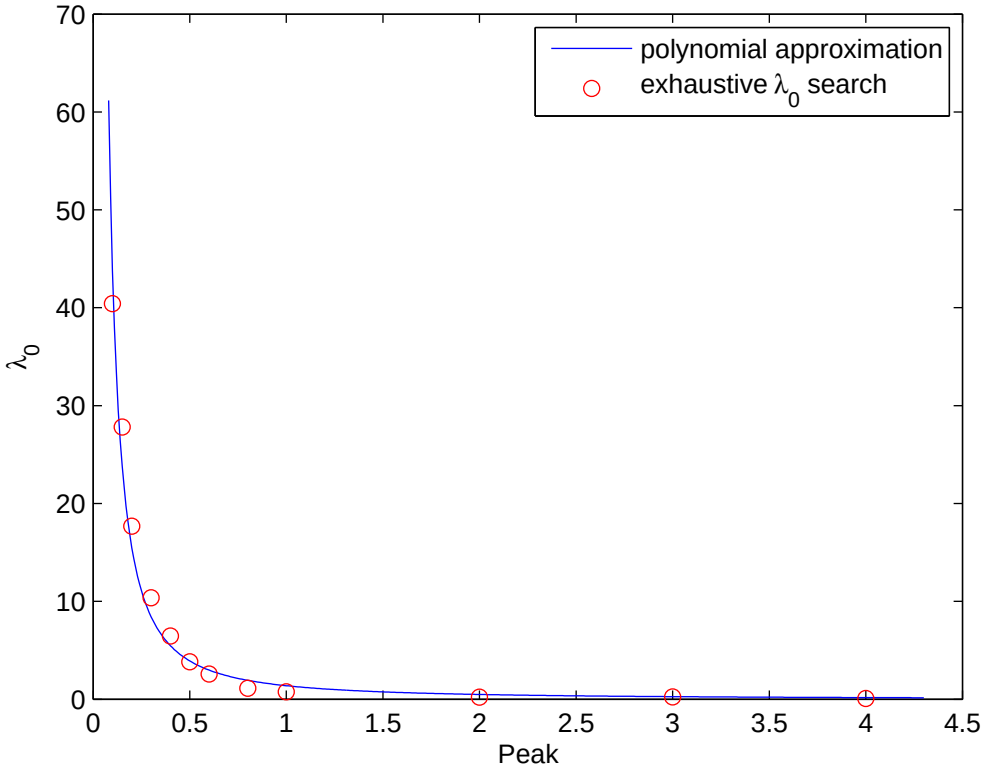


Figure 6.4: Optimal λ_0 and polynomially approximated λ_0 as a function of the peak.

Another important parameter relates to the step size. λ is considered to be proportional to the inverse of the step size, and thus needs to be chosen wisely. We found that increasing λ at each iteration gives better results than using a constant λ. Thus, the update of λ 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. The optimal λ_0 can also be approximated too by a simple polynomial. A good approximation is given by $peak^{-1.5}$. Figures 6.4 Show this visually.

Another important parameter is the number of iterations. We chose to use a fixed number of iterations, but of course this parameter can also be learned or even estimated, similarly to what is done in [RE13], [CBF12]. For denoising we set this value to 50 when binning was used, and to 70 without it. M-P4IP was tested only with binning and the number of iteration is set to 47. In the deblurring scenario we use 60 iterations. Setting the number of iterations is important as the PSNR starts decreasing after a while. This happens because the objective in Equation (4.5) which we try to minimize, 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 peak level, each parameter was tuned for a series of eight images. Out of each original image we generated five degraded images, noisy for the denoising scenario,

and blurred and noisy for the deblurring scenario. We tested multiple peak values in the range of $[0.1, 4]$. We found that the optimal λ_0 and β have strong correlation to 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.

6.2 Denoising

We tested our algorithm on eight different images shown in Figure 6.5

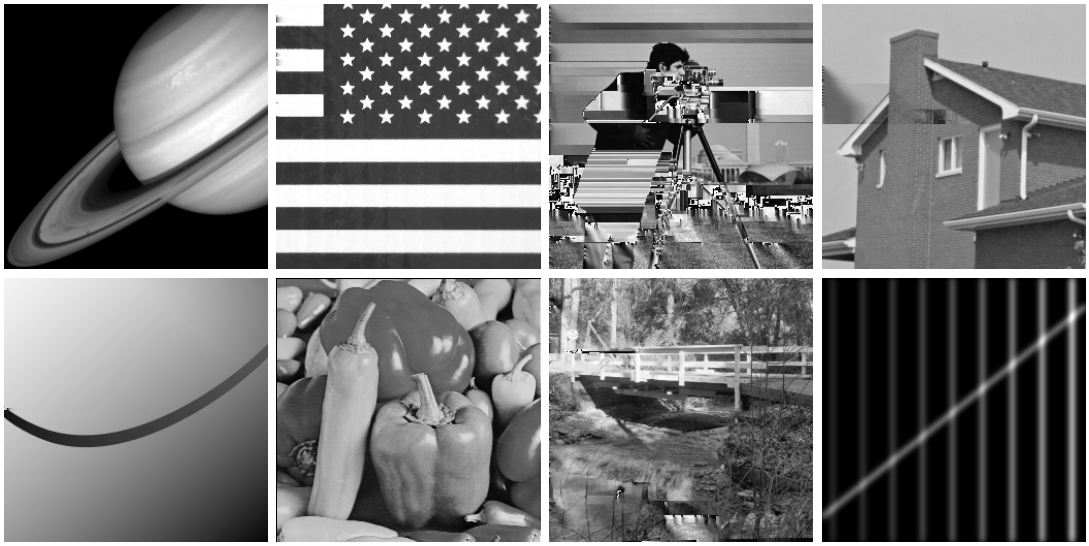


Figure 6.5: Original test images

Out of each image we generated noisy images for peak values 0.1, 0.2, 0.5, 1, 2 and 4. To evaluate our algorithm we compared to BM3D with the refined inverse Anscombe transform [MF11]. We also compared to the SPDA method [GE14], which achieves, to the best of our knowledge, state of the art results. All algorithms were tested with and without binning. The results are shown in Table 6.1, where we have averaged each result over five noisy realizations. While we report results referring to BM3D, we tested several other options (e.g. K-SVD [EA06], WNNM [GZZF14]) and got similar results in spirit and roughly speaking, for the same parameter setting strategy. We eventually chose to report the BM3D ones simply because they were the best in terms of performance vs. computation tradeoff.

Figures 6.7-6.10 present several visual recovery results. Here each PSNR value refers to specific noise realization. As can be seen, the proposed approach competes favorably with both the BM3D+Anscombe and the state of the art results. Binning is found to be beneficial for all algorithms when dealing with low peak values. As for running-times, our algorithm requires roughly 10 sec/image when binning is used, nearly 0.2 seconds took for a single 2nd step run, and negligible time (less then one mili-second) for steps

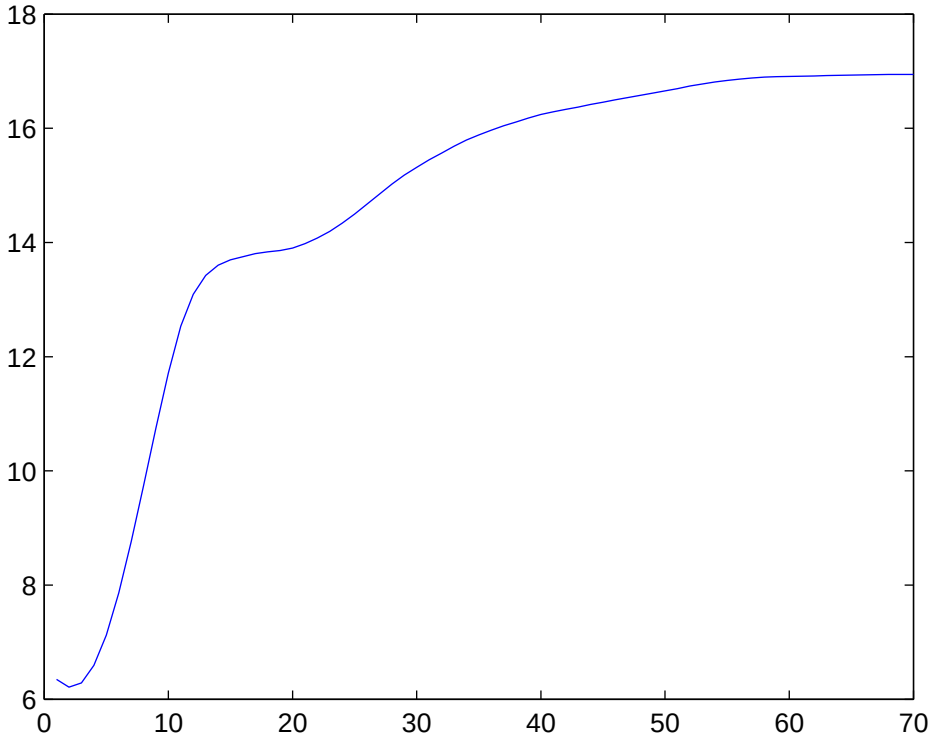


Figure 6.6: PSNR value as a function of the iteration of the image Cameraman for the denoising experiment.

1 and 3. This should be compared to the BM3D+Anscombe that runs somewhat faster (0.2 sec/image), and the SPDA method [GE14] which is much slower (an average of 15-20 minutes/image). When removing the binning, our algorithm runs for one minute, with 0.6 seconds on average for a single 2nd step. BM3D+Anscombe run takes one second and SPDA runs for approximately 10 hours. These runtime evaluations were measured on an i7 with 8GB RAM laptop. Practical convergence of the algorithm on the image "Cameraman" with noise peak 0.1 is shown in Figure 6.6. Due to the modification of the value of λ along the iterations, we get the non-consistent curve as seen in Figure 6.6. Nevertheless, this approach does lead to improved results over the alternative (and more consistent) fixed λ option.

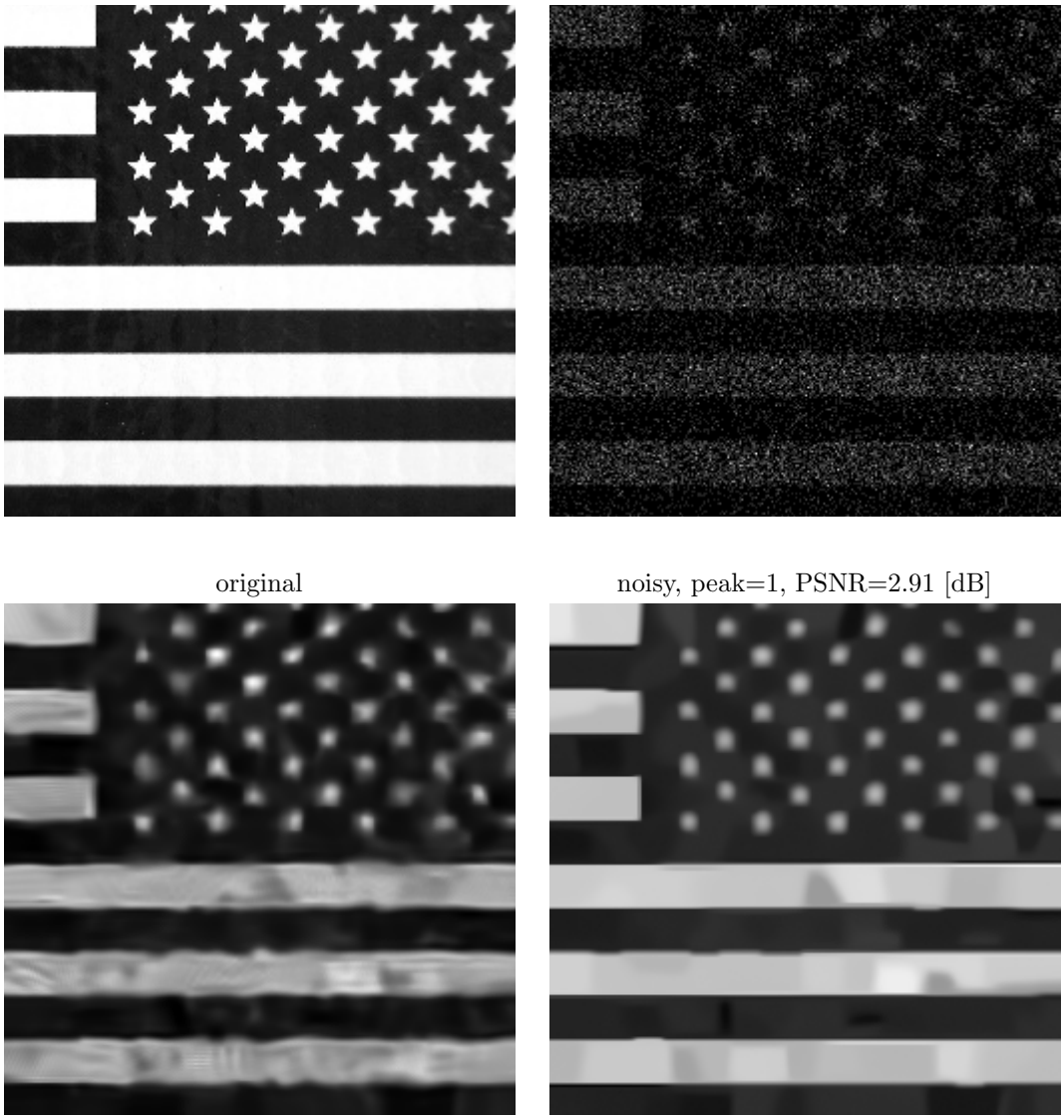
As mentioned before, to check the effectivity M-P^{4}IP we combine BM3D with a simplified version of [SOE14]. We noticed that it is important to find the right prior weight parameter. We only tested for 0.1, 0.2 and 0.5 peaks. The results are shown in table 6.2. For the tested peaks, the multiple prior algorithm shows improvement. We note that it was harder to find good parameters and therefore we believe that it is possible to improve even more.

Table 6.1: Denoising without binning PSNR values [dB]

Alg.	Peak	Saturn	Flag	Cam.	House	Swoosh	Peppers	Bridge	Ridges		Avg
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

Table 6.2: Denoising with binning PSNR values [dB]

Alg.	Peak	Saturn	Flag	Cam.	House	Swoosh	Peppers	Bridge	Ridges		Avg
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



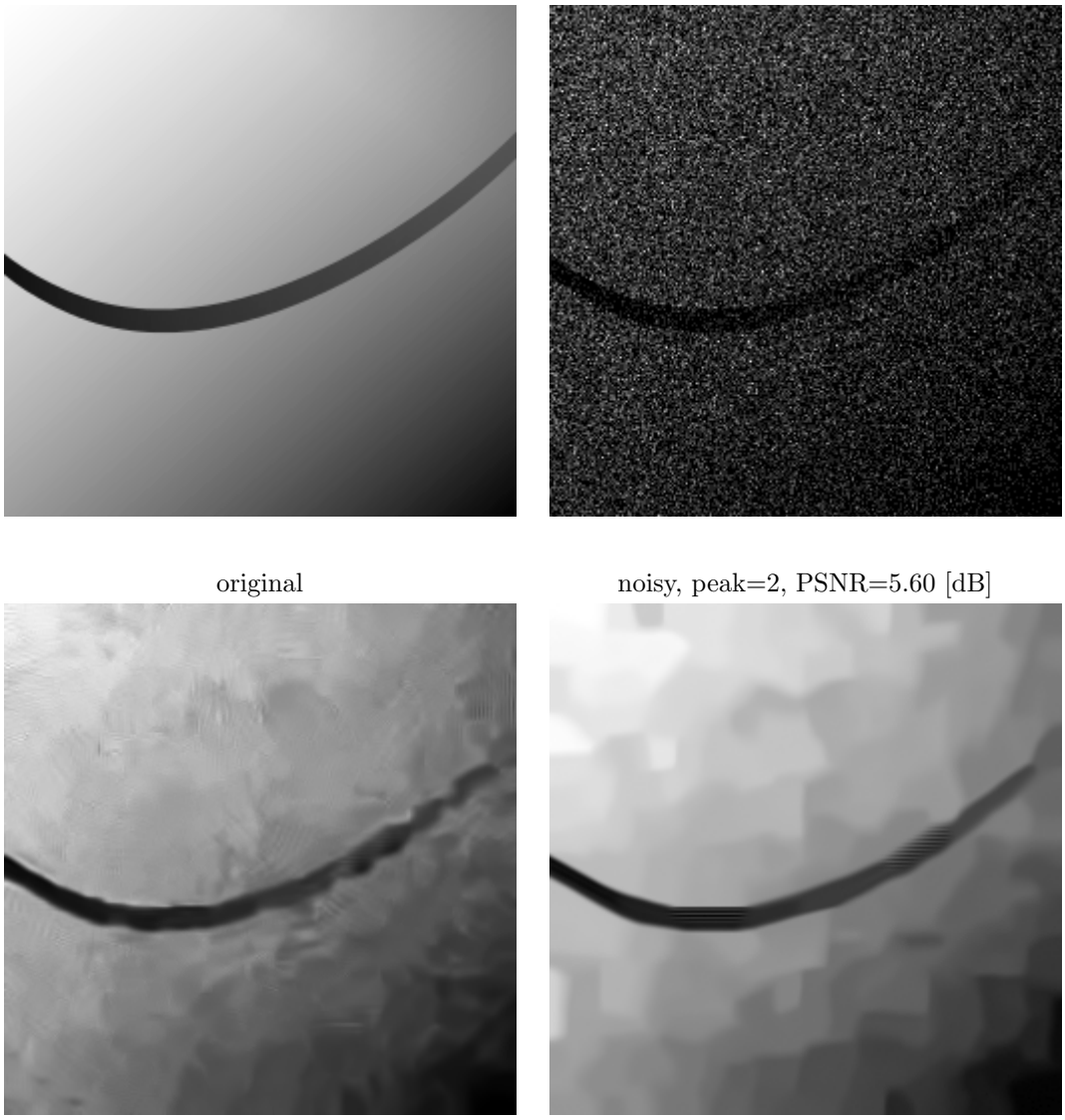
original

noisy, peak=1, PSNR=2.91 [dB]

Anscombe+BM3D, PSNR=18.51

P^4IP, PSNR=18.93 [dB]

Figure 6.7: The image Flag with peak 1 - Denoising (no binning) results.



original

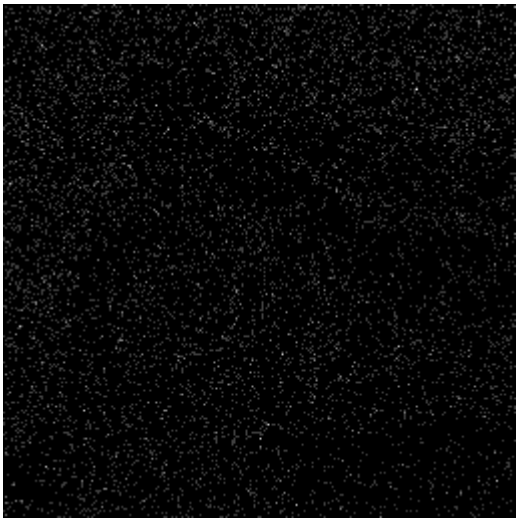
noisy, peak=2, PSNR=5.60 [dB]

Anscombe+BM3D, PSNR=18.52 [dB]

P^4IP, PSNR=29.66 [dB]

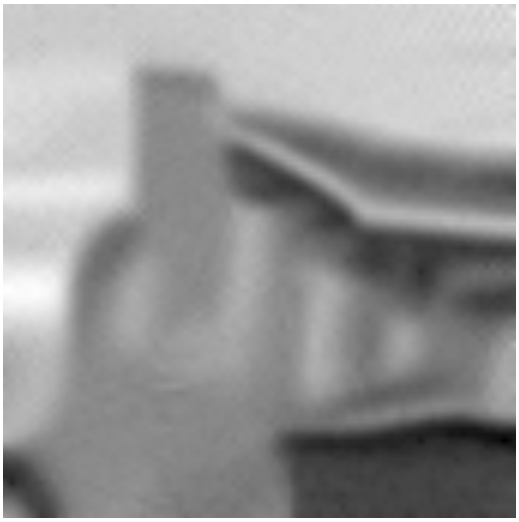
Figure 6.8: The image Ridges with peak 2 - Denoising (no binning) results

original



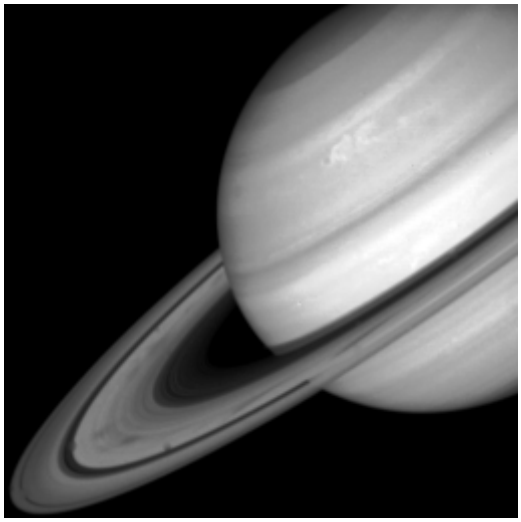
(a) noisy, peak=0.2, PSNR=-4.10 [dB]

Anscombe+BM3D, PSNR=19.90 [dB]

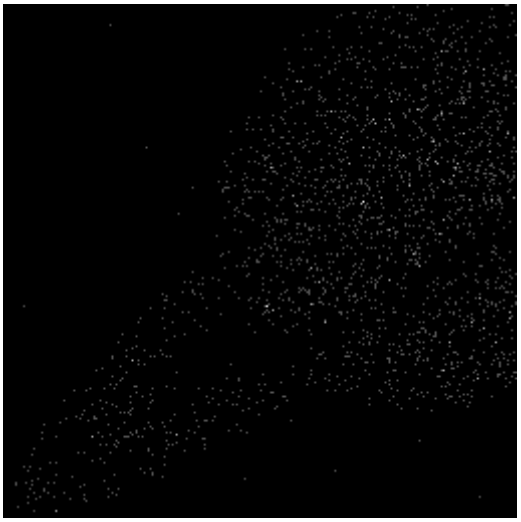


(b) M-P^{4}IP, PSNR=20.18 [dB]

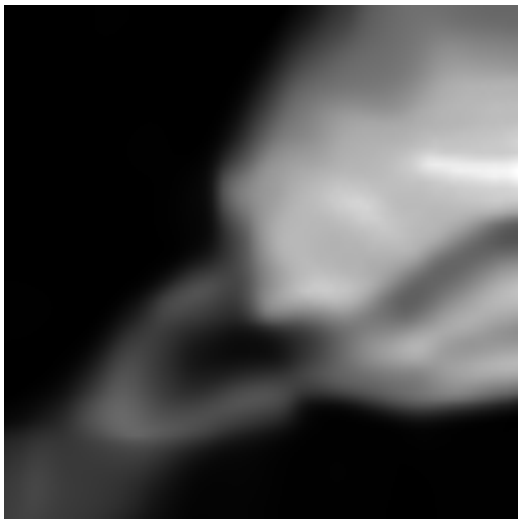
Figure 6.9: The image House with peak 0.2 - Denoising (with binning) results



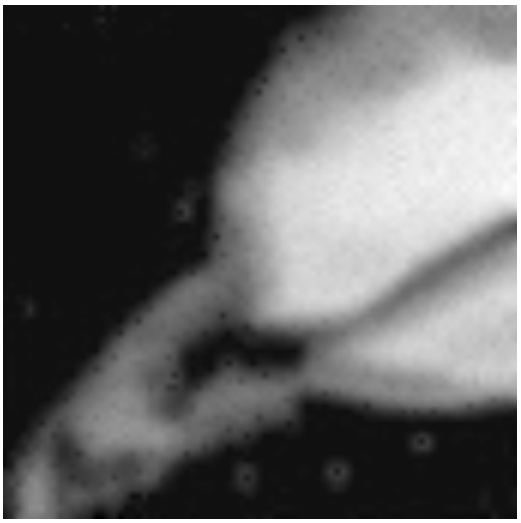
original



(a) noisy, peak=0.1, PSNR=-4.77 [dB]



Anscombe+BM3D, PSNR=20.75 [dB]



(b) M-P^{4}IP, PSNR=21.51 [dB]

Figure 6.10: The image Saturn with peak 0.1 denoising (with binning) results

6.3 Deblurring

In this scenario, we tested our algorithm 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

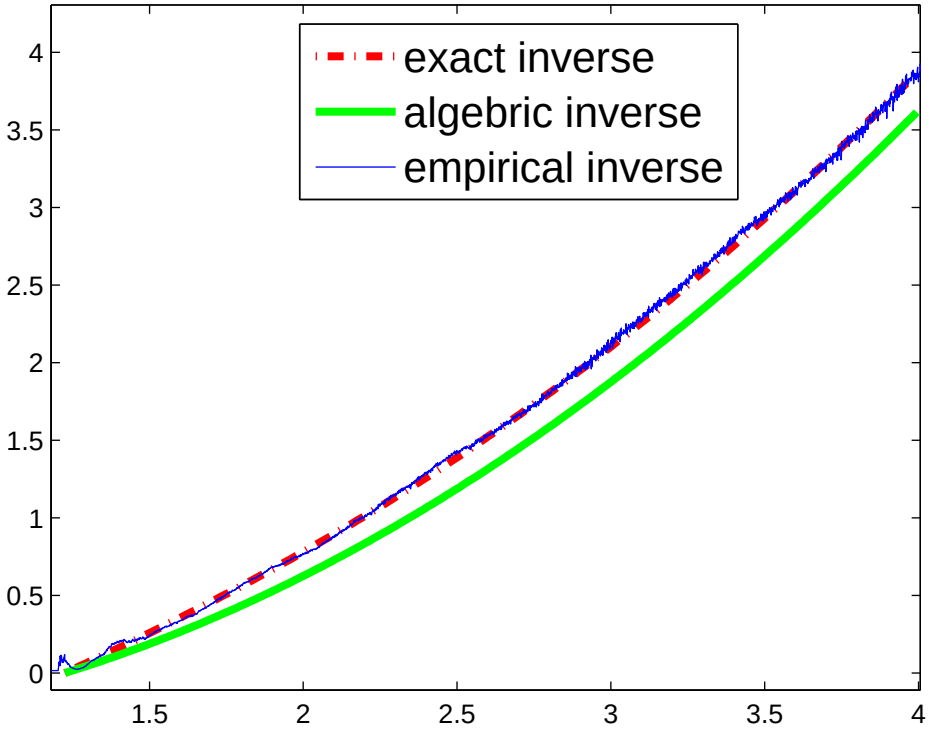


Figure 6.11: Inverse VST curves

To evaluate our algorithm we compared to IDD-BM3D [DKE12] with the refined inverse Anscombe transform [MF11]. It is important to note that such a scheme is essentially inaccurate, because the inversion part is biased. However, 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 Figure 6.11. 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.

We also compared the obtained results to Ma et al. [MMYZ13] which produces state of the art results. This work uses similar ideas as the KSVD algorithm [EA06]. It uses the sparse representations prior with dictionary learning techniques. This work requires an appropriate parameter λ, which we set for each peak, such that it produces the highest average PSNR on all tested images.

The results are shown in Tables 6.3, 6.4 and 6.5. Figures 6.13, 6.14 and 6.15 show specific results to better assess the visual quality of the outcome.

Table 6.3: Deblurring PSNR values for blur kernel (i) [dB]

Method	Peak	Saturn	Flag	Cam.	House	Swoosh	Peppers	Bridge	Ridges		Avg
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^4IP		**26.05**	**18.00**	**19.90**	**21.97**	**26.58**	**19.44**	**18.98**	**26.28**		**22.15**
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^4IP		**27.65**	**19.47**	**20.81**	**23.36**	**28.79**	**20.43**	**19.64**	**28.89**		**23.63**
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^4IP		**29.07**	**20.73**	21.59	**24.83**	30.00	**21.41**	**20.35**	**30.63**		**24.83**

Table 6.4: Deblurring PSNR values for blur kernel (ii) [dB]

Method	Peak	Saturn	Flag	Cam.	House	Swoosh	Peppers	Bridge	Ridges		Avg
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^4IP		**25.41**	**16.98**	**19.54**	**21.52**	**26.08**	**19.00**	**18.66**	**24.54**		**21.47**
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^4IP		**26.83**	**18.52**	**20.18**	**22.46**	**28.42**	19.77	19.21	**26.77**		**22.77**
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^4IP		**28.48**	**20.02**	20.82	**23.72**	**29.47**	20.67	19.76	**29.12**		**24.01**

Table 6.5: Deblurring PSNR values for blur kernel (iii) [dB]

Method	Peak	Saturn	Flag	Cam.	House	Swoosh	Peppers	Bridge	Ridges		Avg
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^4IP		**25.09**	**17.03**	**19.52**	**21.38**	26.21	**19.01**	**18.60**	**23.75**		**21.32**
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^4IP		**26.67**	**18.52**	**20.12**	**22.55**	**28.32**	**19.77**	**19.16**	**25.95**		**22.63**
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^4IP		**27.98**	**20.03**	**20.73**	**23.47**	29.63	**20.57**	**19.75**	**29.08**		**23.91**

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

noise peak 1 is shown in Figure 6.12.

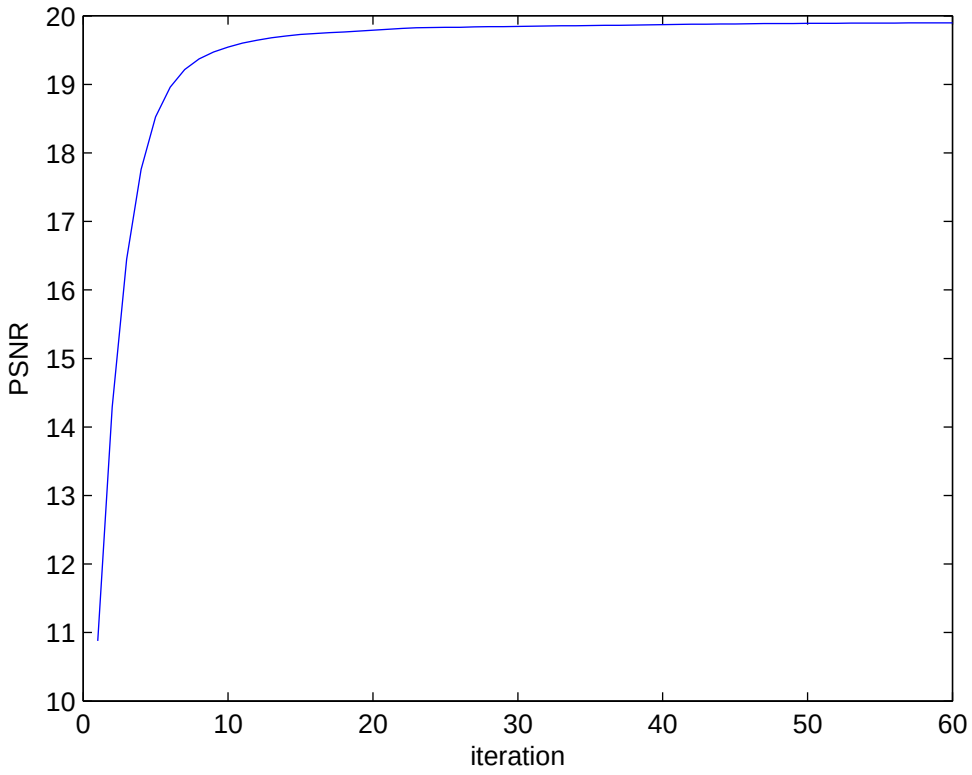
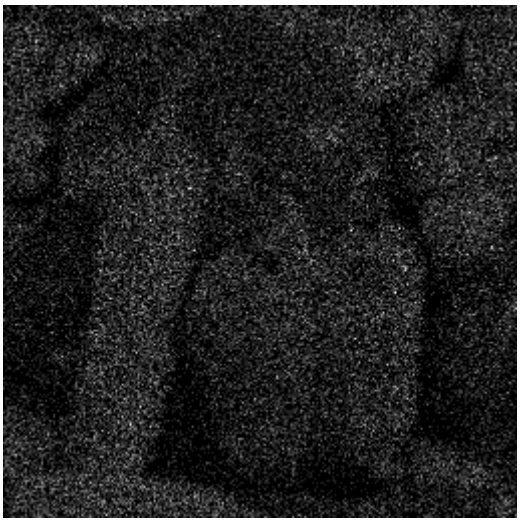


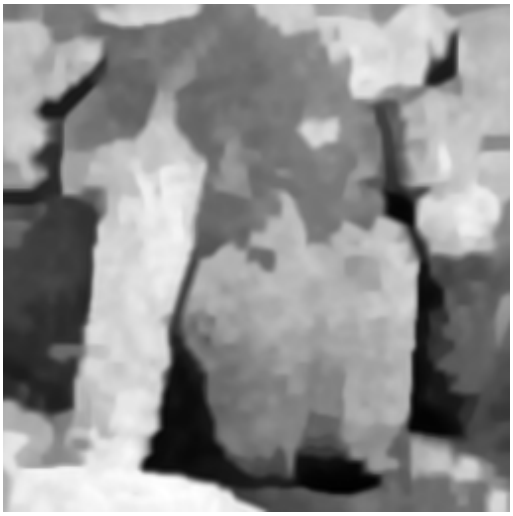
Figure 6.12: PSNR value as a function of the iteration of the image Cameraman for the deblurring experiment.

The runtime for a single image took about 2-3 minutes, with 1.5 seconds for a single 1st step run and with 0.7 seconds 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, the blurring of the images adds degragation. Simply applying the denoising version of P^4IP 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^4IP step which deals with the specific inverse problem should be chosen correctly.

original



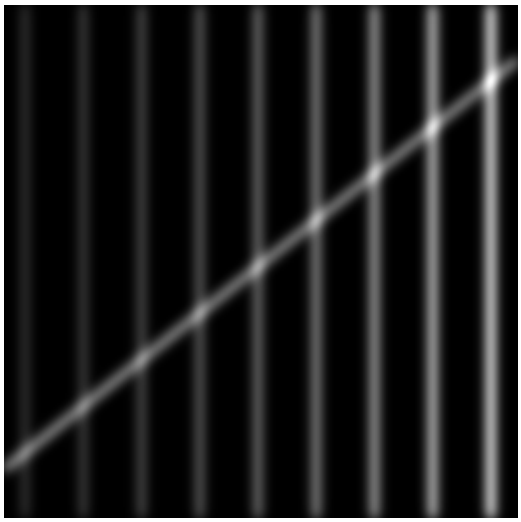
noisy, peak=2, PSNR=6.10 [dB]



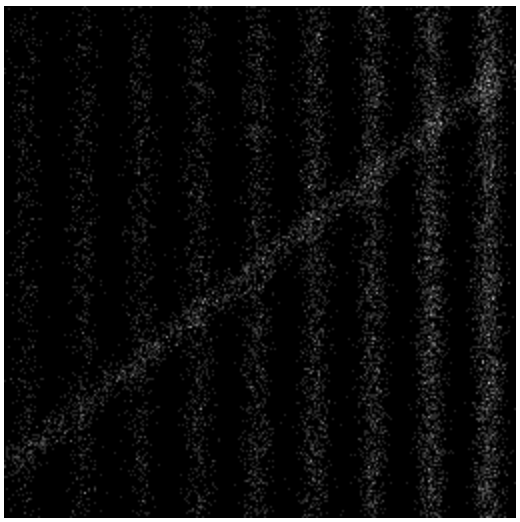
Anscombe with IDD-BM3D,
PSNR=19.72[dB]

P^4IP,
PSNR=20.32[dB]

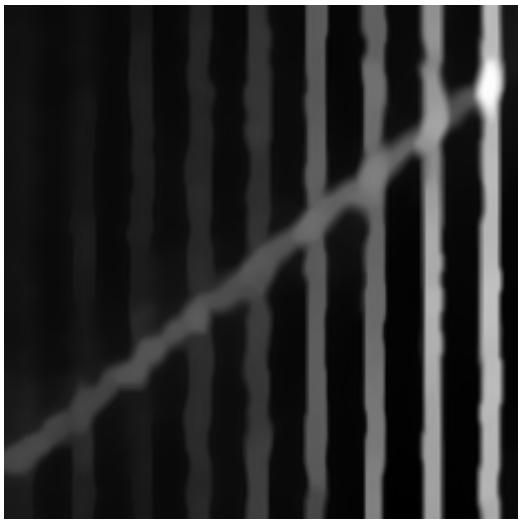
Figure 6.13: The image Peppers with peak 2 and blur kernel (i) - deblurring results.



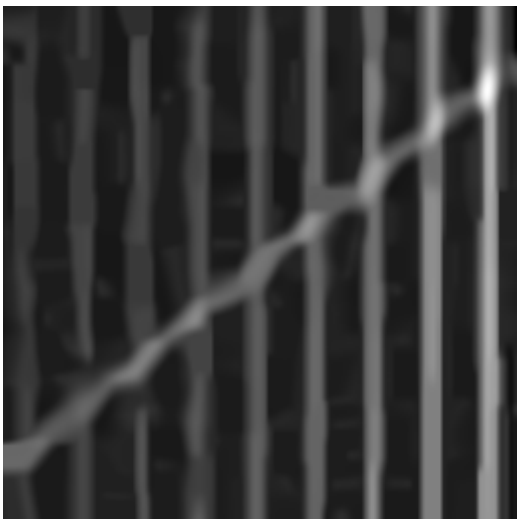
original



degraded, peak=2, PSNR=13.07[dB]



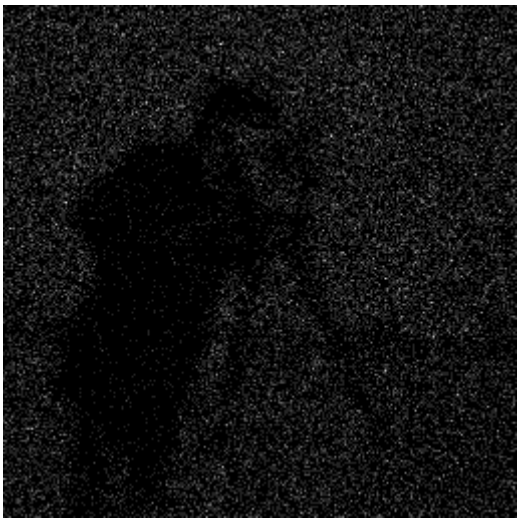
Anscombe with IDD-BM3D,
PSNR=24.04[dB]



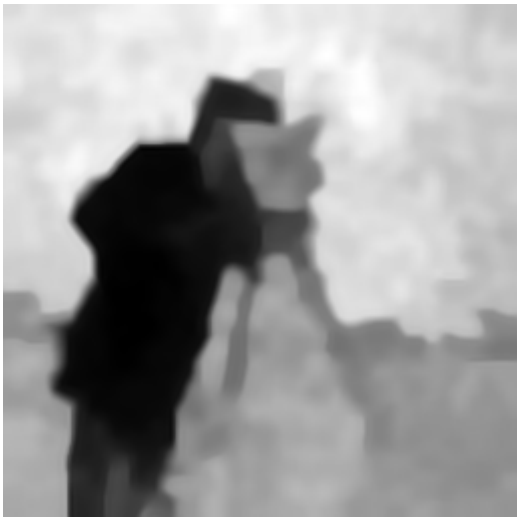
P^{4}IP,
PSNR=26.60[dB]

Figure 6.14: The image Ridges with peak 2 and blur kernel (ii) - deblurring results.

original



degraded, peak=1, PSNR=3.26[dB]



Anscombe with IDD-BM3D,
PSNR=18.97[dB]

P^4IP,
PSNR=19.43[dB]

Figure 6.15: The image Camera Man with peak 1 and blur kernel (iii) - deblurring results.

Chapter 7

Conclusion and Open Questions

This work proposes a new way to integrate Gaussian denoising algorithms to Poisson noise inverse problems, by using the Plug-and-Play Priors 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 - denoising and 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. Possible extension for future work can rely on variants of ADMM [GOSB14],[GTSJ15] that results in a faster PaPP scheme. Many other useful techniques may improve the results. For instance, by applying the P^{4}IP algorithm three times, first with λ_0, second with $\lambda_0 - c$ and third with $\lambda_0 + c$ for some constant value c, and averaging on the three reconstructed images, it is possible to attain a more accurate reconstruction. Of course, this comes at cost of run time. Further work should be done in order to better tune the algorithm's parameters, similar to [DTD10].

Its is also interesting to learn more closely the relation between the Anscombe transform based framework and our method. 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 is possible that more could be said about the 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 [JCPT12]. It would be interesting to see the effectiveness of the PaPP approach for this case.

Appendix A

PaPP for quadratic regularization

In order to get a better understanding of the PaPP framework we look at the following easy problem, where $l(x) = \|x - y\|_2^2$ and $s(x) = \frac{1}{2}\|Ax\|_2^2$. The corresponding regularized problem is

$$\hat{x} = \arg\min_x \|x - y\|_2^2 + \frac{1}{2}\beta \|Ax\|_2^2 . \tag{A.1}$$

The known analytical solution to this problem is

$$x_{analytical} = (\beta A^T A + I)^{-1} y. \tag{A.2}$$

In order to verify that PaPP yields the correct result, we look at the steady state of ADMM's iterations. We note that for the objective function in Equation (A.1), ADMM is guaranteed to converge. The corresponding augmented Lagrangian is given by:

$$L_\lambda = \|x - y\|_2^2 + \frac{1}{2}\beta \|Av\|_2^2 + \frac{\lambda}{2}\|x - v + u\|_2^2 - \frac{\lambda}{2}\|u\|_2^2 . \tag{A.3}$$

This leads to the following ADMM iterations:

$$\begin{aligned} &\text{step 1:} \quad x^{k+1} = \tfrac{y+\lambda v^k - \lambda u^k}{1+\lambda} \\ &\text{step 2:} \quad v^{k+1} = Q \cdot \left(x^{k+1} + u^k\right) \\ &\text{step 3:} \quad u^{k+1} = u^k + x^{k+1} - v^{k+1}, \end{aligned} \tag{A.4}$$

where $Q = \left(\frac{\beta}{\lambda} A^T A + I\right)^{-1}$. At steady state we get:

$$\begin{aligned} x &= \tfrac{y+\lambda v - \lambda u}{1+\lambda} \\ v &= Q \cdot (x + u) \\ u &= u + x - v \end{aligned} \tag{A.5}$$

We can see that the v-update step is actually applying a Gaussian denoising algorithm, which corresponds to the quadratic prior with the appropriate variance. The second

and the third steps give us the relations $x = v$ and $u = (Q^{-1} - I)x$. By plugging this into the first step we have the relation:

$$x = \frac{1}{1+\lambda}y + \frac{\lambda}{1+\lambda}x - \frac{\lambda}{1+\lambda}(Q^{-1} - I)x. \tag{A.6}$$

By plugging the definition of Q we get the correct solution $x_{ADMM} = (\beta A^T A + I)^{-1}y$ as expected.

In order to study the convergence rate we take a similar route to [GTSJ15]. We join the three ADMM steps into one generalized step as follows: First, we use step 2 of Equation (A.4) to get expression for

$$u^k = Q^{-1}v^{k+1} - x^{k+1}. \tag{A.7}$$

Then we plug this expression into step 3 of Equation (A.4) and get

$$u^{k+1} = \left(Q^{-1} - I\right)v^{k+1}. \tag{A.8}$$

Finally, by combining the above with step 1 and 2 of Equation (A.4) we get

$$\begin{aligned} v^{k+1} &= Q \cdot \left(\frac{y + \lambda v^k - \lambda u^k}{1+\lambda} + \left(Q^{-1} - I\right)v^k\right) \\ &= \frac{1}{1+\lambda}Qy + \left(\frac{\lambda - 1}{1+\lambda}Q + \frac{1}{1+\lambda}I\right)v^k, \end{aligned} \tag{A.9}$$

for every k. To find the convergence rate we look at the error between two iteration $e^k = v^k - v^{k-1}$ and get

$$e^{k+1} = E \cdot e^k \tag{A.10}$$

where $E = \frac{\lambda-1}{1+\lambda}Q + \frac{1}{1+\lambda}I$. E's maximal eigenvalue determines PaPP's convergence rate, and therefore we are interested in minimizing it. The derivation of E's eigenvalues is given in theorem A.1.

Theorem A.1. *Let Λ_i be the i-th eigenvalue of $A^T A$, then the i-th eigenvalue of E is given by*

$$\frac{\lambda^2 + \beta\Lambda_i}{\beta\Lambda_i + (1 + \beta\Lambda_i)\lambda + \lambda^2} \tag{A.11}$$

Proof. Let v_i be eigenvector of Q with a corresponding eigenvalue ω_i. then

$$Ev_i = \frac{\lambda - 1}{1+\lambda}Qv_i + \frac{1}{1+\lambda}Iv_i = \left(\frac{\lambda - 1}{1+\lambda}\omega_i + \frac{1}{1+\lambda}\right)v_i \tag{A.12}$$

Therefore v_i is an eigenvector of E with an eigenvalue

$$\frac{\lambda - 1}{1+\lambda}\omega_i + \frac{1}{1+\lambda}. \tag{A.13}$$

because $A^T A$ is symmetric, it is diagonalizable and can be written as $P^T \Lambda P$, where

$P^T P = I$ and Λ is diagonal. we get that Q can be written as $Q = \left(\frac{\beta}{\lambda} P^T \Lambda P + P^T P\right)^{-1} = P\left(\frac{\beta}{\lambda}\Lambda + I\right)^{-1} P^T$, and thus its eigenvalues are $\omega_i = \left(\frac{\beta}{\lambda}\Lambda_i + 1\right)^{-1}$. Plugging this into (A.13) gives

$$\frac{\lambda - 1}{\lambda + 1} \cdot \frac{\lambda}{\beta \cdot \Lambda_i + \lambda} + \frac{1}{\lambda + 1} \tag{A.14}$$

Simple algebra on Equation (A.14) shows that $E's$ eigenvalues are the ones given in Equation (A.11) which concludes our proof. □

Corollary A.2. *The value of λ that minimizes the i's eigenvalue is given by*

$$\lambda_i{}^{opt} = \sqrt{\beta \cdot \Lambda_i} \tag{A.15}$$

The coresponding E's eigenvalue is

$$\frac{2\sqrt{\beta \cdot \Lambda_i}}{\left(\sqrt{\beta \cdot \Lambda_i} + 1\right)^2} \tag{A.16}$$

Proof. Once we know the analytical expression to E's maximal eigenvalue, we can find λ by simply deriving the expression and equating it to 0. Mathematically,

$$\frac{\partial}{\partial \lambda} \frac{\lambda^2 + \beta\Lambda}{\beta\Lambda_i + (1 + \beta\Lambda_i)\lambda + \lambda^2} = 0. \tag{A.17}$$

Simple algebra shows that $\lambda_i{}^{opt} = \sqrt{\beta \cdot \Lambda_i}$. Plugging this into Equation (A.14) concludes our proof. □

Corollary A.3. *We denote λ^{opt} as the value of λ that produces the best convergence rate. Its value is given by:*

$$\lambda^{opt} = \begin{cases} \sqrt{\beta\Lambda_{\max}} \ , \ \frac{\sqrt{\beta\Lambda_{\min}}}{\left(1+\sqrt{\beta\Lambda_{\min}}\right)^2} > \frac{\sqrt{\beta\Lambda_{\max}}}{\left(1+\sqrt{\beta\Lambda_{\max}}\right)^2} \\ \sqrt{\beta\Lambda_{\min}} \ , \ \frac{\sqrt{\beta\Lambda_{\min}}}{\left(1+\sqrt{\beta\Lambda_{\min}}\right)^2} \leq \frac{\sqrt{\beta\Lambda_{\max}}}{\left(1+\sqrt{\beta\Lambda_{\max}}\right)^2} \end{cases} \tag{A.18}$$

This also means that the choice of β effects the convergence rate.

Bibliography

[ABDF10] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo. Fast image recovery using variable splitting and constrained optimization. *Image Processing, IEEE Transactions on*, 19(9):2345–2356, 2010.

[Ans48] F. J. Anscombe. The transformation of poisson, binomial and negative-binomial data. *Biometrika*, pages 246–254, 1948.

[BCM05] A. Buades, B. Coll, and J. M. Morel. A non-local algorithm for image denoising. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 60–65. IEEE, 2005.

[BKB+10] J. Boulanger, C. Kervrann, P. Bouthemy, P. Elbau, J. B. Sibarita, and J. Salamero. Patch-based nonlocal functional for denoising fluorescence microscopy image sequences. *Medical Imaging, IEEE Transactions on*, 29(2):442–454, 2010.

[BPC+11] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[CBF12] M. Carlavan and L. Blanc-Féraud. Sparse poisson noisy image deblurring. *Image Processing, IEEE Transactions on*, 21(4):1834–1846, 2012.

[DFKE07] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *Image Processing, IEEE Transactions on*, 16(8):2080–2095, 2007.

[DFS12] F. X. Dupé, M. J. Fadili, and J. L. Starck. Deconvolution under poisson noise using exact data fidelity and synthesis or analysis sparsity priors. *Statistical Methodology*, 9(1):4–18, 2012.

[DH01] D. L Donoho and X. Huo. Uncertainty principles and ideal atomic decomposition. *Information Theory, IEEE Transactions on*, 47(7):2845–2862, 2001.

[DKE10] A. Danielyan, V. Katkovnik, and K. Egiazarian. Image deblurring by augmented lagrangian with bm3d frame prior. In *Workshop on Information Theoretic Methods in Science and Engineering (WITMSE), Tampere, Finland*, pages 16–18, 2010.

[DKE12] A. Danielyan, V. Katkovnik, and K. Egiazarian. Bm3d frames and variational image deblurring. *Image Processing, IEEE Transactions on*, 21(4):1715–1728, 2012.

[DTD10] C. A. Deledalle, F. Tupin, and L. Denis. Poisson nl means: Unsupervised non local means for poisson noise. In *Image processing (ICIP), 2010 17th IEEE international conference on*, pages 801–804. IEEE, 2010.

[EA06] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, 15(12):3736–3745, 2006.

[Ela10] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer New York, 2010.

[FBD10] M. A. T. Figueiredo and J. Bioucas-Dias. Restoration of poissonian images using alternating direction optimization. *Image Processing, IEEE Transactions on*, 19(12):3133–3145, 2010.

[FN04] P. Fryzlewicz and G. P. Nason. A haar-fisz algorithm for poisson intensity estimation. *Journal of computational and graphical statistics*, 13(3):621–638, 2004.

[GE14] R. Giryes and M. Elad. Sparsity-based poisson denoising with dictionary learning. *Image Processing, IEEE Transactions on*, 23(12):5057–5069, 2014.

[GOSB14] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences*, 7(3):1588–1623, 2014.

[GTSJ15] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson. Optimal parameter selection for the alternating direction method of multipliers (admm): quadratic problems. *Automatic Control, IEEE Transactions on*, 60(3):644–658, 2015.

[GZZF14] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, pages 2862–2869, 2014.

[HN03] J. Hynecek and T. Nishiwaki. Excess noise and other important characteristics of low light level imaging using charge multiplying ccds. *Electron Devices, IEEE Transactions on*, 50(1):239–245, 2003.

[JCPT12] A. Jezierska, E. Chouzenoux, J. C. Pesquet, and H. Talbot. A primal-dual proximal splitting approach for restoring data corrupted with poisson-gaussian noise. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 1085–1088. IEEE, 2012.

[KK04] M. R. Keenan and P. G. Kotula. Accounting for poisson noise in the multivariate analysis of tof-sims spectrum images. *Surface and Interface Analysis*, 36(3):203–212, 2004.

[MBP+09] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2272–2279. IEEE, 2009.

[MF11] M. Makitalo and A. Foi. Optimal inversion of the anscombe transformation in low-count poisson image denoising. *Image Processing, IEEE Transactions on*, 20(1):99–109, 2011.

[MMYZ13] L. Ma, L. Moisan, J. Yu, and T. Zeng. A dictionary learning approach for poisson image deblurring. *Medical Imaging, IEEE Transactions on*, 32(7):1277–1289, 2013.

[PCP11] N. Pustelnik, C. Chaux, and J. C. Pesquet. Parallel proximal algorithm for image restoration using hybrid regularization. *Image Processing, IEEE Transactions on*, 20(9):2450–2462, 2011.

[RE13] Y. Romano and M. Elad. Improving K-SVD denoising by post-processing its method-noise. In *ICIP*, pages 435–439, 2013.

[RSBD08] I. Rodrigues, J. Sanches, and J. Bioucas-Dias. Denoising of medical images corrupted by poisson noise. In *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pages 1756–1759. IEEE, 2008.

[SHDW14] J. Salmon, Z. Harmany, C. A. Deledalle, and R. Willett. Poisson noise reduction with non-local pca. *Journal of mathematical imaging and vision*, 48(2):279–294, 2014.

[SOE14] J. Sulam, B. Ophir, and M. Elad. Image denoising through multi-scale learnt dictionaries. In *Image Processing (ICIP), 2014 IEEE International Conference on*, pages 808–812. IEEE, 2014.

[SSC+10] J. Schmitt, J. L. Starck, J. M. Casandjian, J. Fadili, and I. Grenier. Poisson denoising on the sphere: application to the fermi gamma ray space telescope. *Astronomy & Astrophysics*, 517:A26, 2010.

[SVW+15] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, L. F. Drummy, J. P. Simmons, and C. A. Bouman. Plug-and-play priors for bright field electron tomography and sparse interpolation. *arXiv preprint arXiv:1512.07331*, 2015.

[VBW13] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. 2013.

[YSM12] G. Yu, G. Sapiro, and S. Mallat. Solving inverse problems with piecewise linear estimators: from gaussian mixture models to structured sparsity. *Image Processing, IEEE Transactions on*, 21(5):2481–2499, 2012.

[ZFS08] B. Zhang, J. M. Fadili, and J. L. Starck. Wavelets, ridgelets, and curvelets for poisson noise removal. *Image Processing, IEEE Transactions on*, 17(7):1093–1108, 2008.

ישנו דימיון רב בין ביטוי זה לביטוי התמרת Anscombe. למעשה ניתן להוכיח כי עבור תנאי התחלה מסויימים השונות שמתקבלת זהה עבור שתי ההתמרות. ההבדלים העיקריים בין הגישה שלנו לגישת ה-Anscombe הוא שאצלנו ההתמרה משתנה בכל איטרציה ובנוסף אנו לא מפעילים התמרה הפוכה.

האלגוריתם לפתרון בעיית היפוך כללית המתקבל נקרא אלגוריתם $\mathrm{P}^4\mathrm{IP}$ ופסאודו קוד שלו נתון להלן:

Algorithm A.1 $\mathrm{P}^4\mathrm{IP}$ algorithm

Input: Distorted image y, Gaussian_denoise$(\cdot)$ function

Initialization: set $k = 0$,$u^0 = 0, v^0 =$some initialization;

while !stopping criteria **do**

$x^{k+1} = \arg\min_x -y^T \ln(Hx) + 1^T Hx + \frac{\lambda}{2}\left\|x - v^k + u^k\right\|_2^2$

$v^{k+1} =$ Gaussian_denoise $(x^{k+1} + u^k)$ with $\sigma^2 = \frac{\beta}{\lambda}$

$u^{k+1} = u^k + \left(x^{k+1} - v^{k+1}\right)$

$k = k + 1$

end while

Output: Reconstructed image x^k

גם אם אין ביטוי מפורש לגורם הרגולריזציה. עובדה זו מספקת גמישות רבה בהפעלת הסכימה עם שיטות ניקוי מאוד יעילות.

את סכימת ה־PaPP ניתן להכליל כדי שתעבוד עם מספר ביטויי ידע מקדים. במקרה זה השלבים בכל איטרציה הם:

שלב 1:

$$x^{k+1} = \arg\min_x l(x) + \lambda \left\| x - {v_1}^k + {u_1}^k \right\|_2^2 + \lambda \left\| x - {v_2}^k + {u_2}^k \right\|_2^2.$$

שלב 2:

$${v_1}^k = \arg\min_v \beta_1 s_1(v_1) + \lambda \left\| x^{k+1} - v_1 + {u_1}^k \right\|_2^2$$
$${v_2}^k = \arg\min_v \beta_2 s_2(v_2) + \lambda \left\| x^{k+1} - v_2 + {u_2}^k \right\|_2^2$$

שלב 3:

$${u_1}^{k+1} = u_1^k + x^{k+1} - {v_1}^{k+1}$$
$${u_2}^{k+1} = u_2^k + x^{k+1} - {v_2}^{k+1}$$

גם כאן השלב הראשון הוא פתרון בעיה קמורה, כאשר הביטוי $l(x)$ הוא קמור. השלב השני כולל עתה שני הפעלות של ניקוי רעש גאוסי, והשלב השלישי כולל שני עדכונים פשוטים.

עבור בעיות ההיפוך הפואסוניות, אנו משתמשים בביטוי:

$$l(x) = \ln(P(y|x)) = \sum_i \ln\left(\frac{(Hx)_i^{y_i}}{\Gamma(y_i+1)} e^{-(Hx)_i} \right) = -y^T ln(Hx) + 1^T Hx + constant.$$

מיזעור שלב ה־PaPP הראשון הוא לכן:

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

ניתן כמובן להכליל את הפתרון כך שישתמש במספר אלגוריתמי ניקוי רעש גאוסי.

כאשר אנו מטפלים בבעיית היפוך כללית לא ניתן לפתור את השלב הראשון בצורה אנליטית ויש צורך בשימוש בשיטות מזעור מתחום האופטימיזציה. במקרה של ניקוי רעש פואסוני $H = I$ ואז מיזעור השלב הראשון נהיה פשוט מכיוון שניתן למזער עבור פיקסל בנפרד, והפתרון הוא אנליטי. הביטוי של הפיקסל ה־i ב־x המתקבל הוא:

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

ביטוי זה מהיר מאוד לחישוב.

נמוכה יחסית לרמות האפור של התמונה המקורית. כאשר רמת הרעש גבוהה, התמרת Anscombe לא מבצעת המרה מדוייקת מספיק, ולכן התהליך המתואר מאבד מהאפקטיביות שלו. במקרה כזה יש לטפל בבעיה על ידי טיפול ישיר בסטטיסטיקה הפואסונית, ולא ניתן להשתמש באלגוריתמים הגאוסיים הקיימים.

בעבודה זו, אנו מציעים שיטה חדשה לחיבור אלגוריתמים גאוסיים לתוך בעיית השיחזור הפואסוני, שמבוססת על סכימה כללית שנקראת "Plug and Play Priors" , או PaPP בקיצור. שימוש בסכימה זו מוביל לאלגוריתם איטרטיבי הכולל שלושה שלבים בכל איטרציה. בשלב הראשון, פותרים בעיית מציאת מינימום של פונקציה קמורה. פתרון כזה הוא יעיל מבחינה חישובית. בשלב השני, מפעילים אלגוריתם לניקוי רעש גאוסי, ובשלב השלישי מבצעים עדכון משתנים פשוט. שימוש בשיטה שלנו, בדומה לשיטה המתבססת על התמרת Anscombe, מאפשרת לנצל אלגוריתמים לניקוי רעש גאוסי בצורה קלה ויעילה. למרות הדמיון הרב בין שתי השיטות, הבסיס המתמטי עליו בנויה השיטה שלנו הוא שונה, ורלוונטי לכל טווחי רמות הרעש. אנו מציגים שתי אפליקציות הבנויות על השיטה שלנו – ניקוי רעש (denoising) פואסוני, ותיקון טשטוש (deblurring) של תמונה הסובלת מרעש פואסוני. בשני המקרים אנו משיגים תוצאות טובות יותר מאשר שיטות המתבססות התמרות Anscombe ודומיהן ואף משיגים תוצאות דומות לשיטות ישירות מובילות.

בשיטת ה־PaPP מנסים לבצע שיערוך MAP ע"י מיזעור

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

נשתמש בסימונים הבאים: $l\left(x\right) = -ln\left(P\left(y|x\right)\right)$ וגם $s\left(x\right) = -ln\left(P\left(x\right)\right)$.

נסתכל על הבעיה השקולה מתמטית הבאה:

$$\hat{x} = \underset{x,v\in R^{m\times n}}{\arg\min}\ l\left(x\right) + \beta s\left(v\right).$$
$$s.t\ \ x = v$$

כאשר הוספנו פרמטר β כדי לקבל גמישות באלגוריתם. ניתן לפתור בעיה זו ע"י שיטת ה-ADMM. על פי שיטה זו אנו צריכים לבצע שלושה שלבים.

$$\begin{aligned}
x^{k+1} &= \arg\min_{x} l\left(x\right) + \tfrac{\lambda}{2}\left\|x - \left(v^k - u^k\right)\right\|_2^2, \\
v^{k+1} &= \arg\min_{v} \tfrac{\lambda}{2}\left\|x^{k+1} + u^k - v\right\|_2^2 + \beta s\left(v\right), \\
u^{k+1} &= u^k + \left(x^{k+1} - v^{k+1}\right).
\end{aligned}$$

בשלב הראשון אנו פותרים בעיה קמורה קלה לפתרון. השלב הזה תלוי בבעית השיחזור אותה אנו מנסים לפתור. עבור בעיית ניקוי רעש פואסוני, שלב זה ניתן לפתרון בצורה אנליטית פשוטה. השלב השני הוא הפעלה של אלגוריתם לניקוי רעש גאוסי על תמונת הביניים $x^{k+1} + u^k$. בדרך זו אנו משתמשים בידע המוקדם של האלגוריתם הגאוסי שאיתו אנו עובדים. השלב השלישי הוא שלב עדכון פשוט וקל למימוש. מובטח כי התהליך יתכנס למינימום לוקאלי של הפונקציה אותה אנו מנסים למזער. נקודה חשובה בסכימה המוצעת הינה שניתן להפעיל אלגוריתם נתון להורדת רעש גאוסי

תקציר

בבעיות היפוך אנו מקבלים תמונה שעברה קילקול ומנסים לשחזר ממנה את התמונה הנקייה המקורית. ישנם סוגים רבים של קילקולים שתמונה יכולה לעבור כגון: הוספת רעש לתמונה, טשטוש תמונה, הקטנת תמונה ועוד. דרך שהוכיחה את יעילותה במשימות שיחזור אלו היא שימוש בשיטות שיערוך סטטיסטיות, הטובות בעיקר במקרים שאנו יודעים את הסטטיסטיקה של הרעש שקילקל את התמונה. משערך סטטיסטי נפוץ מאוד נקרא משערך MAP המנסה למצוא את התמונה x שבהסתברות הגבוהה ביותר יצרה את התמונה המקולקלת הנתונה y. מתמטית, אנו מחפשים:

$$\hat{x} = \arg\max_x P\left(x|y\right) = \arg\max_x \frac{P\left(y|x\right)P\left(x\right)}{P\left(y\right)} = \arg\max_x P\left(y|x\right)P\left(x\right).$$

בהינתן x, כלומר התמונה אותה אנו מנסים לשחזר, ידוע לנו הקשר הסטטיסטי $P(y|x)$. הידע נובע מהעובדה שאנו יודעים את סטטיסטיקת הרעש. עבור בעיית הרעש הפואסוני הביטוי $P(y|x)$ נתון ע"י:

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

כאשר $x[i]$ קובע את הממוצע של ההתפלגות של הפיקסל ה־i.

הביטוי $P(x)$ מתאר את ההסתברות שתמונה x מייצגת תמונה אמיתית. זהו למעשה הידע המקדים שלנו על אילו מאפיינים יש לכל תמונה בעולם. ביטוי זה אינו תלוי בבעית ההיפוך שאנו פותרים. אלגוריתמים חזקים לשיחזור תמונות מתבססים על ידע מקדים טוב, דבר שרלוונטי עבור בעיות נוספות.

דרך אחת לפיתרון בעיות היפוך פואסוניות היא שימוש בהתמרת Anscombe. התמרה זו הינה התמרה הממירה משתנה מקרי פואסוני למשתנה מקרי גאוסי בעל שונות קבועה. טרנספורמציה זו מעניינת מכיוון שהיא קלה לחישוב ופותחת דלת לפיתרון של בעיות היפוך עם רעש פואסוני. אופן הפתרון מורכב משלושה שלבים. תחילה מתבצעת הפעלה של ההתמרה, שנית, מופעל אלגוריתם המיועד לשיחזור תמונה עם רעש גאוסי, ולבסוף מופעלת ההתמרה ההופכית. דרך זו שימושית עקב קיומם של אלגוריתמי שיחזור חזקים המיועדים עבור רעש גאוסי. אלגוריתמים אלו הוכיחו את עצמם בשלל אפליקציות הן מבחינה תיאורטית והן מבחינה מעשית.

התהליך המתואר נמצא יעיל עבור תמונות בעלות יחס אות לרעש גבוה, כלומר עוצמת הרעש הינה

המחקר בוצע בהנחייתו של פרופסור מיכאל אלעד, בפקולטה למדעי המחשב.

אני מודה לטכניון על התמיכה הכספית הנדיבה בהשתלמותי.

פתרון בעיות היפוך פואסוניות בעזרת סכימה לחיבור ידע מקדים

חיבור על מחקר

לשם מילוי חלקי של הדרישות לקבלת התואר
מגיסטר למדעים במדעי המחשב

אריה רונד

הוגש לסנט הטכניון – מכון טכנולוגי לישראל
אייר התשע"ו חיפה מאי 2016

פתרון בעיות היפוך פואסוניות בעזרת סכימה לחיבור ידע מקדים

אריה רונד