Mixed Poissonian-Gaussian Denoising Using ADMM

Nicholas Tan
Stanford University
ntan2012@stanford.edu

Abstract

This study assesses the use of a fast optimization algorithm (ADMM) on the relevant problem of denoising in raw image sensor data. The focus of the study is on solving the case of a mixture of Poissonian and Gaussian based noise, for which there is no straightforward analytical solution for the proximal operator in ADMM. However, by considering the two prior distributions separately, we can find an analytical solution for each. We propose a signal-based mask to integrate the two solutions. It is also shown that this coupling of the two algorithms yields better results than using a pure Poissonian prior.

1. Introduction

1.1. Motivation and Background

Noise in image sensors has often been modeled as either purely Gaussian or purely Poissonian. Gaussian noise can be used to approximate camera read noise; and Poissonian noise can approximate temporal shot noise. However, there are many settings where sensors may operate in a state where both noise sources are present. Moreover, at different intensities, the weighting of each distribution will be different. Explicitly, at lower intensity regions of the sensor dynamic range, Gaussian noise begins to dominate; and conversely, at higher intensity regions of the dynamic range, Poissonian noise is the main contributor to measurement uncertainty. [1]

1.2. Related Work

The literature on this subject is extensive, given that this is an understood and prevalent issue in most state-of-the-art camera systems. Significantly, there has already been work utilizing proximal algorithms with good results [2], outperforming our own method. Their approach involves a pre-processing step that transforms the Poissonian noise to an approximate standard Gaussian using the Anscombe transform.

There are also examples of other effective methods such as using a Hidden Markov Model [3] or other generalized methodologies such as PURE-LET image deconvolution [4].

For analyzing purely Poissonian denoising, there has also been work done. Significantly, the ADMM process has been formalized and analytical solutions are understood and have been implemented. [5]

2. Methods

2.1. Variable Definitions

I : the ideal image, before any alterations

b : the input image, after blurring and with Poissonian-Gaussian noise added

x : the estimated output image

λ : hyper-parameter controlling influence of regularization on the optimization

ρ : augmented Lagrangian hyper-parameter controlling the step size of the dual-variable update

k : soft-thresholding hyper-parameter, k = $\frac{\lambda}{\rho}$

2.2. Alternating Direction Method of Multipliers

We utilize a classical approach to efficiently solving distributed convex optimization systems. Alternating Direction Method of Multipliers (henceforth, ADMM) is especially fast when dealing with larger datasets, such as that of images, which often scale to millions of pixels per single image data-point. This method decomposes a larger minimization problem into several subproblems and solves each using proximal operations. In its most general form, we can write the ADMM equation as follows:

$$\min_{\{x\}} f(x) + g(z)$$

$$\text{subject to } Kx - z = 0$$

for some $f, g, K,$ and z.

To make our implementation more efficient, we can provide function handles that perform image-wide parallel operations instead of costly matrix point calculations. ADMM

may be further optimized by performing all convolution-like operations in the Fourier domain, utilizing the Convolution Theorem, which reduces the complexity of each operation. Lastly, there is a significant amount of pre-computation that is possible which saves on valuable in-loop computations.

2.3. Poissonian Deconvolution Using ADMM

The standard form of ADMM, applied to our image formulation, is as follows:

$$\min_{\{x\}} g_1(z_1) + g_2(z_2)$$
$$\text{subject to } Kx - z = 0$$

In this case, we can codify the Poissonian prior distribution into the equation such that minimization of $g_1(z_1)$ will lead to a maximum likelihood estimate. Explicitly, given the prior distribution p, we would like to minimize the negative log likelihood $g_1(z_1) = -\log(p(b|z_1))$.

We also apply an additional non-negativity constraint on the data during the optimization process. Explicitly, this is codified in the penalty function $g_2(z_2) = I_{R+}(z_2)$ where

$$I_{R+}(z_2) = \begin{cases} 0 & z_2 \in \mathbb{R}+ \\ \infty & z_2 \notin \mathbb{R}+ \end{cases}$$

K then becomes

$$K = \begin{bmatrix} A \\ I \end{bmatrix}$$

and z becomes

$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

We can then write the Augmented Langrangian and solve for the iterative updates as follows:

```
for i = 1:max_iters

     v = z - u;
     numer = fft(c)'*fft(v1)+fft(v2);
     denom = fft(c)'*fft(c)+1;
     x = ifft(numer/denom);

     v = A*x + u1;
     b_2a = (1-pv)/2/p;
     z1 = -b_2a+sqrt(b_2a^2+b/p);

     v = x + u2;
     z2 = max(0, v);

     u = u + K*x - z;

end
```

Here, the x update is the closed-form solution for the proximal operator of the quadratic subproblem, which aims to minimize the sum of squares between z and Kx. Pre-computation is possible for the denominator and fft(c) in the numerator for this calculation. The z1 update is the closed form solution to maximum likelihood of the data for Poisson distributed noise. Note that each pixel is updated independently here. The z2 update is to enforce the non-negativity constraints on the image.

2.4. Poissonian Deconvolution with TV prior

In this section, we update the above algorithm to include a common prior used in digital image processing denoising applications - total variation regularization. This prior preserves important detail features such as edges while penalizing unwanted noisy variations in the image. Formally, we express these additional constraints as extra terms in the standard ADMM process as follows:

$$\min_{\{x\}} g_1(z_1) + g_2(z_2) + g_3(z_3) + g_4(z_4)$$
$$\text{subject to } Kx - z = 0$$

Here, g_1 and g_2 are the same as before :
$g_1(z_1) = -\log(p(b|z_1))$ and $g_2(z_2) = I_{R+}(z_2)$
The additional terms will express the regularization :
$g_3 = \lambda\|z_3\|_1$ and $g_4 = \lambda\|z_4\|_1$
K and z will also be updated and extended to:

$$K = \begin{bmatrix} A \\ I \\ D_x \\ D_y \end{bmatrix}, z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix}$$

The Augmented Langragian is similar to the pure Poissonian ADMM formulation. Solving for the iterative updates, we get:

```
for i = 1:max_iters

     v = z - u;
     numer = fft(c)'*fft(v1)+fft(v2)+
           fft(dx)'*fft(v3)+fft(dy)'*fft(v4);
     denom = fft(c)'*fft(c)+1+
           fft(dx)'*fft(dx)+fft(dy)'*fft(dy);
     x = ifft(numer/denom);

     v = A*x + u1;
     b_2a = (1-pv)/2/p;
     z1 = -b_2a+sqrt(b_2a^2+b/p);

     v = x + u2;
     z2 = max(0, v);

     v = dx*x + u3;
```

```
    z3 = Sk(v)

    v = dy*x + u4;
    z4 = Sk(v)

    u = u + K*x + z;

end
```

Note here that K and z are defined as above and Sk represents the soft-thresholding operator, which has a piecewise function as follows:

$$\text{Sk}(v) = \begin{cases} v - k & v > k \\ 0 & |v| \leq k \\ v + k & v < -k \end{cases}$$

where k = $\frac{\lambda}{\rho}$

Again, we can optimize the above algorithm by precomputing the denominator of the fft of the x update. The z1 and z2 updates enforce, as before, the Poissonian prior and non-negativity constraints, respectively. The z3 and z4 updates are to enable the total variation prior.

2.5. Gaussian Deconvolution with TV prior

Gaussian ADMM assumes that there is a constant addition of noise independently across each pixel of the image i.e. $b = \|Ax + \eta\|$ where η is chosen from some 0-mean normal distribution parametrized by its standard deviation. We can formally express this desire as:

$$\min_{\{x\}} f(x) + g_2(z_2) + g_3(z_3) + g_4(z_4)$$
$$\text{subject to } Kx - z = 0$$

where $f(x) = \frac{1}{2}\|Ax - b\|_2^2$,
and, as before, $g_2(z_2) = I_{R+}(z_2)$
$g_3 = \lambda\|z_3\|_1$ and $g_4 = \lambda\|z_4\|_1$
and

$$K = \begin{bmatrix} I \\ D_x \\ D_y \end{bmatrix}, z = \begin{bmatrix} z_2 \\ z_3 \\ z_4 \end{bmatrix}$$

Solving for the proximal operators associated with these terms ultimately leads to the iterative updates as follows:

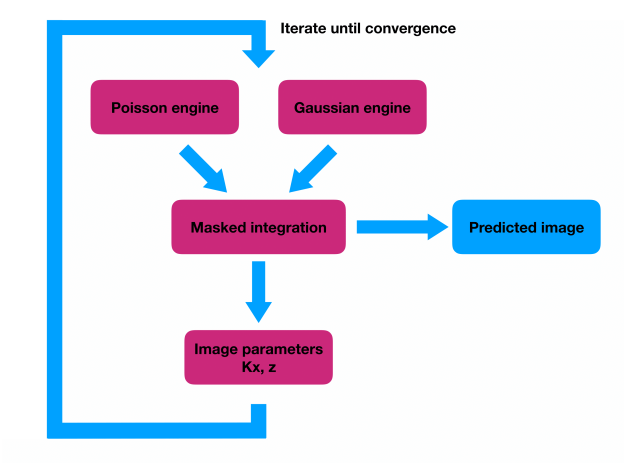
```
for i = 1:max_iters

    v = z - u;
    numer = fft(c)'*fft(b)+fft(v2) +
          p*( fft(dx)'*fft(v3) +
              fft(dy)'*fft(v4) );
    denom = fft(c)'*fft(c)+1    +
          p*( fft(dx)'*fft(dx) +
              fft(dy)'*fft(dy) );

    x = ifft(numer/denom);

    v = x + u2;
    z2 = max(0, v);

    v = dx*x + u3;
    z3 = Sk(v)

    v = dy*x + u4;
    z4 = Sk(v)

    u = u + K*x + z;

end
```

The x-update arises as the analytical solution to the normal equations coming from the result of the proximal operator. It includes one term that minimizes the expected Gaussian noise. The z2 update enforces the non-negativity constraints. The z3 and z4 updates are regularization terms that enforce the total-variation prior. They use the soft-thresholding operator Sk as before.

Again, we can optimize the run-time of the algorithm by precomputing the denominator and known input image terms. We also perform the calculation in the Fourier domain to avoid costly point-wise matrix computations.

2.6. Integrating Poissonian-Gaussian ADMM

To mix the two iterative algorithms, we first must calibrate where in the dynamic range the transition between Poissonian dominated noise and Gaussian dominated noise occurs. Knowing this, we can construct a weight-mask that will be applied to our images during each iteration of the optimization. Using the threshold from calibration, we compute a sigmoid centered at this value that will distribute the influence of the Poisson estimation against the influence of the Gaussian estimation. For pixels with higher intensity than the threshold, the sigmoid allocates more weight to the Poissonian estimation, and vice-versa.



Here we show an example mask calculation:

From left to right: original image, noisy image, mask

We note that the mask is a contrast-enhanced copy of the noisy image. Effectively, the mask will clip lighter areas and force those into the Poissonian domain as well as clip darker areas and force those into the Gaussian domain. To make transitions between the two domains spatially smoother, we apply an image filter to this mask:

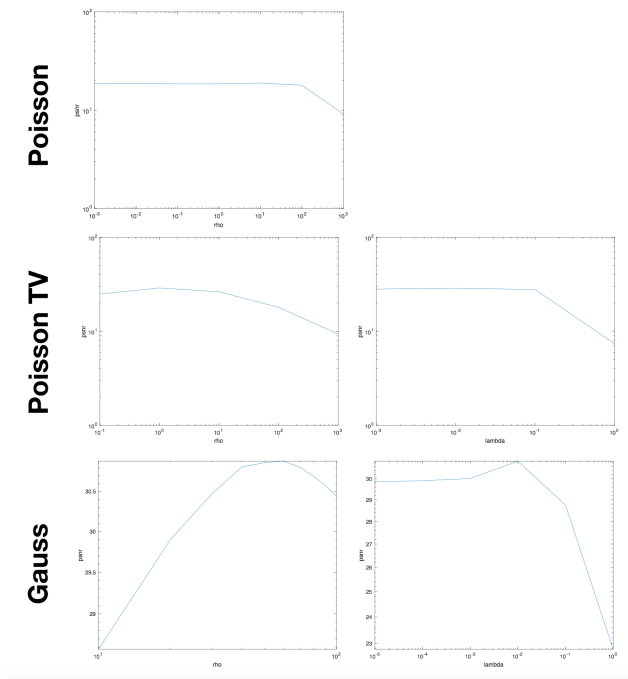
Once the mask has been calculated, we can apply it to the Poissonian-estimated image and its inverse to the Gaussian-estimated image. These are then combined and fed into the image parameter calculations for the next iteration. Specifically, the updates to z2, z3, and z4 are done with the integrated image to enforce the non-negativity and sparse total-variation constraints.

Some code is provided below:

```
for i = 1:max_iters

     v = z - u;
     numer = fft(c)'*fft(b)+fft(v2) +
           p*( fft(dx)'*fft(v3) +
               fft(dy)'*fft(v4) );
     denom = fft(c)'*fft(c)+1   +
           p*( fft(dx)'*fft(dx) +
               fft(dy)'*fft(dy) );
     x_gauss = ifft(numer/denom);

     v = z - u;
     numer = fft(c)'*fft(b)+fft(v2) +
           p*( fft(dx)'*fft(v3) +
               fft(dy)'*fft(v4) );
     denom = fft(c)'*fft(c)+1   +
           p*( fft(dx)'*fft(dx) +
               fft(dy)'*fft(dy) );
     x_poiss = ifft(numer/denom);

     v = A*x + u1;
     b_2a = (1-pv)/2/p;
     z1 = -b_2a+sqrt(b_2a^2+b/p);

     x = w*x_poiss + (1-w)*x_gauss;

     v = x + u2;
     z2 = max(0, v);

     v = dx*x + u3;
     z3 = Sk(v)

     v = dy*x + u4;
     z4 = Sk(v)

     u = u + K*x + z;

end
```

2.7. Optimization of hyper-parameters

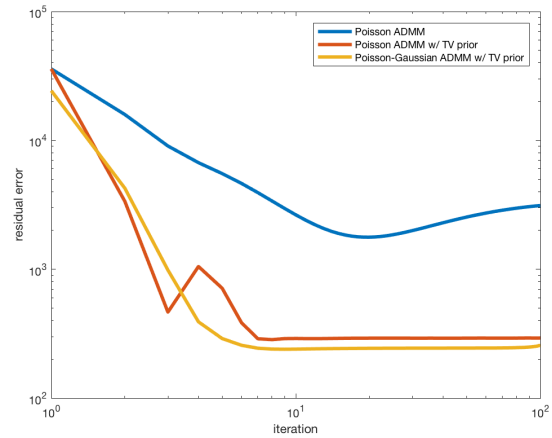
To choose the values of ρ and λ, we do a 2D parametric sweep and choose the pair that generates the highest PSNR.



Without doing this calibration, we see severe ringing and diagonal image artifacts from the choice of horizontal and vertical kernels that perform the total-variation evaluation.

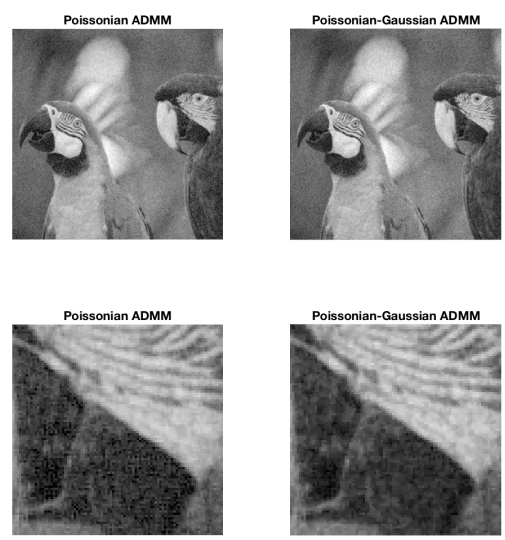
3. Results

We see better results with our algorithm, combining the influence of the Poissonian terms and Gaussian terms. Using the residual error from the ideal image as a metric to evaluate performance, we see improvement when mixing the two distributions compared to using ADMM based on purely Poissonian noise. The residual error was calculated at each iteration of the optimization as: $\frac{1}{2}\|I - x\|_2^2$



We also used PSNR as a related, second metric to evaluate performance. Poissonian-Gaussian denoising yields higher PSNR's across the board (see Appendix).

The algorithm yields especially favorable results in the darker areas of the image, where Gaussian noise begins to dominate:



In the darker areas, there is less noise present and the signal is smoother and more uniform overall. Moreover, the high level detailed structure is still maintained with Poissonian-Gaussian denoising.

4. Conclusion

In the presence of mixed Poissonian-Gaussian noise, using an ADMM algorithm that accounts for the difference in prior distributions yields the best results. Because the noise is signal dependent, we are able to construct a signal-based mask to relatively weight the likelihood of noise appearing from each distribution. To maintain speed, we separately

calculate the analytical solutions to the proximal operators in ADMM with a Gaussian-based model and a Poissonian-based model, integrating their image estimates using the mask. This method is able to improve PSNR by 1-2dB over the pure Poissonian denoising ADMM. Given more time, future work would include implementing related algorithms and comparing results.

5. References

[1] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical Poissonian-Gaussian Noise Modeling and Fitting for Single-Image Raw-Data," IEEE Transactions on Image Processing, vol. 17, no. 10, pp. 17371754, 2008.

[2] Mohamed Aly and Wolfgang Heidrich, "MindX: Denoising Mixed Impulse Poisson-Gaussian Noise Using Proximal Algorithms"

[3] Sejung Yang and Byung-Uk Lee, "Poisson-Gaussian Noise Reduction Using the Hidden Markov Model in Contourlet Domain for Fluorescence Microscopy Images"

[4] Florian Luisier, Thierry Blu, and Michael Unser, "Image Denoising in Mixed PoissonGaussian Noise"

[5] Mrio A. T. Figueiredo and Jos M. Bioucas-Dias, "Restoration of Poissonian Images Using Alternating Direction Optimization"


PSNR=21.0309
PSNR=16.5727
PSNR=24.6845
PSNR=26.5286
PSNR=20.2444
PSNR=15.4576
PSNR=24.0059
PSNR=25.2331
PSNR=20.9947
PSNR=16.1874
PSNR=26.4939
PSNR=27.2365
PSNR=20.8651
PSNR=16.4924
PSNR=23.0087
PSNR=24.2996
PSNR=20.6909
PSNR=15.8952
PSNR=25.934
PSNR=26.9993
Original
Poissonian-Gaussian Noise added
Poissonian ADMM
Poissonian ADMM w/TV
Poissonian-Gausian ADMM w/TV