Using a Gaussian Denoiser for a Mixed Poisson/Gaussian Noise

Dorian Baudry – Paul Chevalier

4 avril 2019

Introduction

- Context : even though noise present in satellite optical images is a mixture of Poisson and Gaussian noise, the great majority of denoisers work only in the case of Gaussian noise.
- We studied two ways to circumvent this inadequacy :
  - Gaussian approximation of the noise and use of a Gaussian Denoiser after stabilizing its variance
  - Plug in the Gaussian denoiser in an algorithm that uses the knowledge of the distribution of the noise

What are the advantages and drawbacks of these methods ?

An example using a synthetic noise

Table of contents

Mixed Poisson Gaussian Noise

Let's denote x an image without any noise and y a noisy observation. For GP noise :

$$y = \alpha p + n \tag{1}$$

With $p \sim \mathcal{P}(x)$ and $n \sim \mathcal{N}(0, \sigma^2)$ independent. The moments are :

$$\begin{aligned} \mathbb{E}(y|x) &= \alpha x \\ \mathbb{V}(y|x) &= \alpha^2 x + \sigma^2 \end{aligned} \tag{2}$$

Gaussian approximation : $y/\alpha \sim \mathcal{N}(x, x + \sigma^2/\alpha^2)$

Some existing Gaussian Denoisers

Nowadays state of the art Denoising algorithms mainly fall in two categories :

- Bayesian Methods using self-similarities between patches : NL-Bayes
- Neural Networks with CNN layers : DnCNN, FFDNet

We used the IPOL implementation of the DnCNN algorithms in our experiments

Using Gaussian Denoiser on Transformed input

To use the Gaussian denoiser we need to remove the luminence dependancy of the noise, by applying a Variance Stabilizing Transform. The denoising algorithm is :

- Input y, set $z = 2\sqrt{y/\alpha + (\sigma/\alpha)^2} = g_\alpha(y/\alpha)$
- Denoise z
- Return $g_\alpha^{-1}(z)$

The transformation is obtained by using a Taylor expansion of a function f at y/α and setting $f'(x) = 1/\sqrt{x + (\sigma/\alpha)^2}$.

This transform ensures $\mathbb{V}(g_\alpha(y/\alpha)) = C$ for some real C

The ADMM scheme

Idea : Compute the MAP of the underlying image :

$$\hat{x} = \text{argmax}_x\{-log(p(y|x)) - log(p(x))\} \tag{3}$$

Problem : We don't have any prior of the clean image x, So we replace it by any regularization function g and set :

$$\hat{x} = \text{argmax}_x\{f(x) + \lambda g(x)\} \tag{4}$$

The algorithm introduces a slack variable v and a constraint $x = v$, and consider the augmented Lagrangian function :

$$\mathcal{L}(x, v, u) = f(x) + \lambda g(v) + u^T(x - v) + \frac{\rho}{2}||x - v||^2 \tag{5}$$

Iteratively solving the ADMM problem

We pass the details to provide the steps of the ADMM denoising algorithm :

- Set $x^{(k+1)} = \text{argmin } f(x) + \rho^{(k)}/2 \times ||x - v^{(k)} + u^{(k)}||^2$
- Set $v^{(k+1)} = D_{\sigma_k}(x^{(k+1)} + u^{(k)})$, D is a denoiser
- Set $u^{(k+1)} = u^{(k)} + x^{(k+1)} - v^{(k+1)}$
- Update $\rho^{(k)}$, $\sigma^{(k)} = \sqrt{\lambda/\rho^{(k)}}$

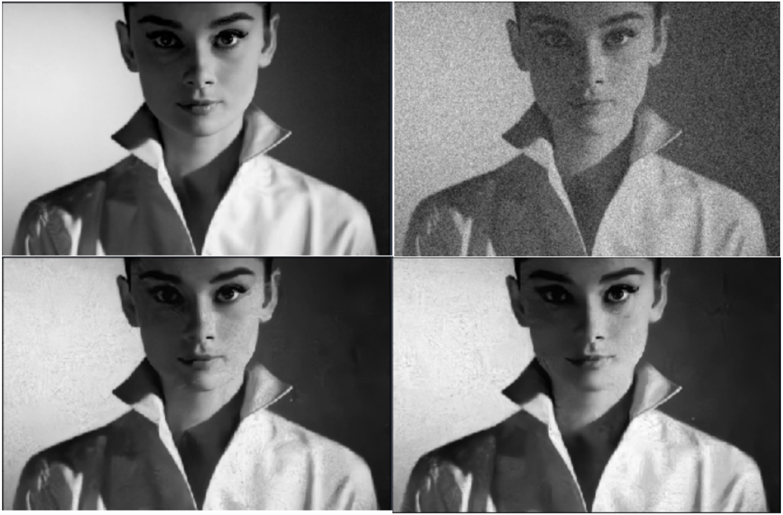
The initial values of x, v and u can be arbitrary. The parameter ρ is updated according to a *continuation scheme*.

ρ, λ and the continuation rules need to be **properly tuned**.

The steps are repeated **until convergence**.

A first experiment

A second experiment



Conclusion of our experiments

- VST is fast and provides quite good results for all sets of parameters. Denoising perf similar as with Gaussian noise.
- ADMM method also provides good results but may be computationally expensive (Maximization step for each pixel and slow convergence)

VST often provides a better PSNR than ADMM, but this is balanced by the fact the visual quality of the denoised image is generally better for ADMM : ***The choice of the algorithm may depend on the application***

Last remark : both algorithms are quite unstable on the noise parameters.