Journal of Computational and Applied Mathematics 343 (2018) 373–396

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

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

Accelerating $\ell^1 - \ell^2$ deblurring using wavelet expansions of operators

Paul Escande [a,*], Pierre Weiss [b]

[a] Département d'Ingénierie des Systèmes Complexes (DISC), Institut Supérieur de l'Aéronautique et de l'Espace (ISAE), Toulouse, France
[b] Institut des Technologies Avancées en Sciences du Vivant, ITAV-USR3505 and Institut de Mathématiques de Toulouse, IMT-UMR5219, CNRS and université de Toulouse, Toulouse, France

ARTICLE INFO

Article history:
Received 9 March 2017
Received in revised form 10 August 2017

MSC:
65F50
65R30
65T60
65Y20
42C40
45Q05
45P05
47A58

Keywords:
Sparse wavelet expansion
Preconditioning
GPU programming
Image deblurring
Inverse problems

ABSTRACT

Image deblurring is a fundamental problem in imaging, usually solved with computationally intensive optimization procedures. The goal of this paper is to provide new efficient strategies to reduce computing times for simple deblurring models regularized using orthogonal wavelet transforms. We show that the minimization can be significantly accelerated by leveraging the fact that images and blur operators are compressible in the same orthogonal wavelet basis. The proposed methodology consists of three ingredients: (i) a sparse approximation of the blur operator in wavelet bases, (ii) a diagonal preconditioner and (iii) an implementation on massively parallel architectures. Combining the three ingredients leads to acceleration factors ranging from 4 to 250 on a typical workstation. For instance, a 1024×1024 image can be deblurred in 0.15 s.

1. Introduction

Most imaging devices produce blurry images. This degradation very often prevents the correct interpretation of image contents and sometimes ruins expensive experiments. One of the most advertised examples of that type is Hubble space telescope,[1] which was discovered to suffer from severe optical aberrations after being launched. Such situations occur on a daily basis in fields such as biomedical imaging, astronomy or conventional photography.

Starting from the seventies, a large number of numerical methods to deblur images were therefore developed. The first methods were based on linear estimators such as the Wiener filter [2]. They were progressively replaced by more complicated nonlinear methods, incorporating prior knowledge on the image contents. We refer the interested reader to the following review papers [3–5] to get an overview of the available techniques.

Despite providing better reconstruction results, the most efficient methods are often disregarded in practice, due to their high computational complexity, especially for large 2D or 3D images. The goal of this paper is to develop new numerical

* Corresponding author.
E-mail addresses: paul.escande@gmail.com (P. Escande), pierre.armand.weiss@gmail.com (P. Weiss).

[1] The total cost of Hubble telescope is estimated at 10 billions US Dollars [1].

https://doi.org/10.1016/j.cam.2018.04.063
0377-0427/

strategies that significantly reduce the computational burden of image deblurring. The proposed ideas yield a *fast deblurring method*, compatible with *large data* and *routine use*. They allow handling both *stationary and spatially varying blurs* with an identical complexity. The proposed algorithm does not reach the state-of-the-art in terms of image quality, because the prior is too simple, but still performs well in short computing times. A real-time deblurring experiment is provided here https://www.youtube.com/watch?v=oHnGNRc9Qeo&feature=youtu.be.

1.1. Image formation and image restoration models

In this paper, we assume that the observed image u_0 reads:

$$u_0 = Hu + b, \tag{1}$$

where $u \in \mathbb{R}^N$ is the clean image we wish to recover, $b \sim \mathcal{N}(0, \sigma^2 I_N)$ is a white Gaussian noise of standard deviation σ and $H \in \mathbb{R}^{N\times N}$ is a known blurring operator. Loosely speaking, a blurring operator replaces the value of a pixel by a mean of its neighbors. A precise definition will be given in Section 4.

Let $\Psi \in \mathbb{R}^{N\times N}$ denote an orthogonal wavelet transform and let $A = H\Psi$. A standard variational formulation to restore u consists of solving:

$$\min_{x\in\mathbb{R}^N} E(x) = F(x) + G(x). \tag{2}$$

In this equation, $F(x)$ is a quadratic data fidelity term defined by

$$F(x) = \frac{1}{2}\|Ax - u_0\|_2^2. \tag{3}$$

The regularization term $G(x)$ is defined by:

$$G(x) = \|x\|_{1,w} = \sum_{i=1}^{N} w[i]|x[i]|. \tag{4}$$

The vector of weights $w \in \mathbb{R}_+^N$ is a regularization parameter that may vary across sub-bands of the wavelet transform. The weighted ℓ^1-norm is well known to promote sparse vectors. This is usually advantageous since images are compressible in the wavelet domain. Overall, problem (2) consists of finding an image Ψx consistent with the observed data u_0 with a sparse representation x in the wavelet domain.

Many groups worldwide have proposed minimizing similar cost functions in the literature, see e.g. [6–9]. The current trend is to use frames Ψ such as the undecimated wavelet transforms or learned transforms instead of orthogonal transforms [10–13]. This usually allows reducing reconstruction artifacts. We focus here on the case where Ψ is orthogonal. This property will help designing much faster algorithms traded for some image quality.

1.2. Standard optimization algorithms

A lot of algorithms based on proximity operators were designed in the last decade to solve convex problems of type (2). We refer the reader to the review papers [14,15] to get an overview of the available techniques. A typical method is the accelerated proximal gradient descent, also known as FISTA (Fast Iterative Soft Thresholding Algorithm) [16]. By letting $\|A\|_2$ denote the largest singular value of A, it takes the form described in Algorithm 1. This method got very popular lately due to its ease of implementation and relatively fast convergence.

Algorithm 1 Accelerated proximal gradient descent

1: **input:** Initial guess $x^{(0)} = y^{(1)}$, $\tau = 1/\|A\|_2^2$ and Nit.
2: **for** $k = 1$ to Nit **do**
3: Compute $\nabla F(y^{(k)}) = A^*(Ay^{(k)} - u_0)$. ▷ **99.35″**
4: $x^{(k)} = \mathrm{Prox}_{\tau G}\left(y^{(k)} - \tau\nabla F(y^{(k)})\right)$. ▷ 2.7″
5: $y^{(k+1)} = x^{(k)} + \frac{k-1}{k+2}(x^{(k)} - x^{(k-1)})$. ▷ 1.1″
6: **end for**

Let us illustrate this method on a practical deconvolution experiment. We use a 1024×1024 image and assume that $Hu = h \star u$, where $\star$ denotes the discrete convolution product and h is a motion blur described in Fig. 4b. In practice, the deblurred image stabilizes visually after 500 iterations. The computing time on a workstation with Matlab and mex-files is around 103″15. The result is shown in Fig. 5. Profiling the code leads to the computing times shown on the right-hand-side of Algorithm 1. As can be seen, 96% of the computing time is spent in the gradient evaluation. This requires computing two wavelet transforms and two fast Fourier transforms. This simple experiment reveals that two approaches can be used to reduce computing times: