An Iterative Linear Expansion of Thresholds for ℓ_1-based Image Restoration

Hanjie Pan, *Student Member, IEEE* and Thierry Blu, *Fellow, IEEE*

***Abstract*—This paper proposes a novel algorithmic framework to solve image restoration problems under sparsity assumptions. As usual, the reconstructed image is the minimum of an objective functional that consists of a data fidelity term and an ℓ_1 regularization. However, instead of estimating the reconstructed image that minimizes the objective functional directly, we focus on the restoration process that maps the degraded measurements to the reconstruction. Our idea amounts to parameterizing the process as a linear combination of few elementary thresholding functions (LET) and solve for the linear weighting coefficients by minimizing the objective functional. It is then possible to update the thresholding functions and to iterate this process (*i*-LET). The key advantage of such a linear parametrization is that the problem size reduces dramatically—each time we only need to solve an optimization problem over the dimension of the linear coefficients (typically less than 10) instead of the whole image dimension. With the elementary thresholding functions satisfying certain constraints, global convergence of the iterated LET algorithm is guaranteed. Experiments on several test images over a wide range of noise levels and different types of convolution kernels clearly indicate that the proposed framework usually outperform state-of-the-art algorithms in terms of both CPU time and number of iterations.**

***Index Terms*—Image restoration, sparsity, majorization minimization (MM), iterative reweighted least square (IRLS), thresholding, linear expansion of thresholds (LET).**

I. INTRODUCTION

A. Problem Description

Consider the standard image restoration problem: find a good estimation of the original image $\mathbf{x}$ from the degraded measurements $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$. Here $\mathbf{H}$ is an $M \times N$ matrix, which models the linear mapping between the original image $\mathbf{x} \in \mathbb{R}^N$ and the measurements $\mathbf{y} \in \mathbb{R}^M$, and $\mathbf{n} \in \mathbb{R}^M$ is an additive noise corruption. Specifically, $\mathbf{H}$ can model the point spread function (PSF) in an optical lens system, the Radon transform in tomographic reconstruction, or missing pixels for inpainting problems, etc. For many practical problems, $\mathbf{H}$ is ill-conditioned or non-invertible, which precludes the possibility to apply direct inverse filtering to the measurements. Therefore prior knowledge of the image is required to regularize the restoration problem [2], [3]. An often used recent approach is to regularize the inverse problem with the sparse-promoting ℓ_1 norm in some transformation domain (see [4]–[6]), giving rise to the constrained formulation of the problem as:

$$\min_{\mathbf{c}\in\mathbb{R}^D} \|\mathbf{c}\|_1 \quad \text{s. t.} \quad \|\mathbf{y} - \mathbf{H}\mathbf{W}\mathbf{c}\|_2 \leq \varepsilon, \tag{1}$$

where $\mathbf{c} \in \mathbb{R}^D$ is the transformation coefficients and the image $\mathbf{x} = \mathbf{W}\mathbf{c}$. Specifically, for wavelet-based image restoration problems, $\mathbf{W} \in \mathbb{R}^{N\times D}$ represents the orthogonal or tight frame wavelet synthesis, $\mathbf{W}^{\mathrm{T}}$ is the corresponding wavelet analysis, and ε is a non-negative parameter that is related to the noise variance (e.g. $\varepsilon = \sqrt{M}\sigma$ for an M-pixel distorted image with noise variance σ^2). When $\varepsilon = 0$, (1) reduces to the well-known *basis-pursuit* [7]. Another closely related constrained formulation of the problem optimizes the data-fidelity term $\|\mathbf{y} - \mathbf{H}\mathbf{W}\mathbf{c}\|_2^2$ under the constraint that the ℓ_1 norm of the corresponding transformation coefficients is bounded:

$$\min_{\mathbf{c}\in\mathbb{R}^D} \|\mathbf{y} - \mathbf{H}\mathbf{W}\mathbf{c}\|_2^2 \quad \text{s. t.} \quad \|\mathbf{c}\|_1 \leq \varepsilon, \tag{2}$$

which is known as *least absolute shrinkage and selection operator* (LASSO [8]). Detailed survey of various algorithms to solve LASSO and its relationships with *basis-pursuit* can be found in [9]–[11]. Arguably, instead of addressing the constrained problems (1) and (2) directly, most algorithms deal with the Lagrangian of the constrained problem:

$$J(\mathbf{c}) = \overbrace{\|\mathbf{y} - \mathbf{H}\mathbf{W}\mathbf{c}\|_2^2}^{J_0(\mathbf{c})} + \lambda\|\mathbf{c}\|_1, \tag{3}$$

$$\hat{\mathbf{c}} = \arg\min_{\mathbf{c}\in\mathbb{R}^D} J(\mathbf{c}), \tag{4}$$

where $\lambda \geq 0$ is the regularization weight, which balances the data fidelity and the sparsity of the reconstruction. The restored image is reconstructed as $\hat{\mathbf{x}} = \mathbf{W}\hat{\mathbf{c}}$. Alternatively but not equivalently, the regularization is directly applied to the image:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}\in\mathbb{R}^N} \left\{\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + \lambda\|\mathbf{W}^{\mathrm{T}}\mathbf{x}\|_1\right\}, \tag{5}$$

which is known as the *analysis formulation*. The most popular analysis formulation is probably the total-variation (TV) based image restoration [12], [13]. When redundant transformation $\mathbf{W}$, $\mathbf{W}^{\mathrm{T}}$ is used, solutions to the synthesis and analysis formulation problems are generally different [5], [6], [14], [15]. We are not going to discuss which formulation is preferable in this paper but will focus on the minimization of the unconstrained problem (3) in the following sections. Yet, we give a very efficient algorithm in Section IV-D that allows to solve for problem (1) (i.e. find λ in problem (3)) within the iterations of our proposed approach.

Copyright © 2013 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

The material was presented in part at IEEE International Conference on Image Processing (ICIP'11) at Brussels, Belgium, September 2011 [1].

The authors are with the Department of Electronic Engineering, the Chinese University of Hong Kong (e-mail: hanjie.pan@ieee.org; thierry.blu@m4x.org).

B. Approaches to solve the problem

Notice that minimization of (3) can be cast as least square programming with linear constraints. Thus, in principle it can be solved via standard convex optimization algorithms, such as interior point (IP) method. However, the computational complexity for most image processing problems rules out complicated interior point methods, which require explicit access to each components of $\mathbf{H}$ and $\mathbf{H}^{\mathrm{T}}$ matrix. However, see [16] for a recent development that deploys interior point method with preconditioning and makes it applicable to image processing. Other approaches that are able to handle large scale problems are investigated and developed, including gradient methods [17], [18], parallel coordinate descend (PCD) methods [5], fixed-point continuation methods [19], gradient projection methods [20], proximal methods [21]–[25] and iterative shrinkage thresholding (IST) methods [4], [26].

The IST is probably one of the most popular algorithms to solve (4) because of its simplicity. It requires one matrix multiplication of $\mathbf{H}$ and $\mathbf{H}^{\mathrm{T}}$, which can be efficiently implemented in most practical image restoration problems, followed by the shrinkage:

$$\mathbf{c}^{(n+1)} = \theta_{\lambda\tau/2}\Big(\mathbf{c}^{(n)} - \tau\mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\big(\mathbf{H}\mathbf{W}\mathbf{c}^{(n)} - \mathbf{y}\big)\Big),$$

where $\theta_T(t) = \mathrm{sign}(t)\max\left(|t| - T, 0\right)$ is the soft- threshold function and τ is the IST step-size. IST was first introduced in the framework of *expectation maximization* (EM) [27]. It was then reinterpreted in the more general *majorization minimization* (MM) settings [28] by majorizing the Hessian of the quadratic data-fidelity term in the objective functional.

However, it is also recognized that IST is a slow-converging algorithm. In [29], [30], it is proven that IST has a *linear* global convergence rate and that under certain conditions, it can converge arbitrarily slowly. Another drawback of IST-like algorithms is that they require a warm start for faster convergence: if the algorithm is poorly initialized such that it is too far away from the final solution, then it may require much more iterations to converge. Typically, IST is initialized with arbitrarily small values that are different from zeros. Such an initialization is based on the prior knowledge that the final solution is sparse and hence many coefficients are close to zero. Another way to better initialize IST is related to the continuation scheme [19], [20] that speeds up the convergence rate, which first solves the minimization problem for a larger λ in (3) before gradually decreasing it to the desired λ in the subsequent steps.

Recent efforts to speed up IST has led to the emergence of algorithms like TwIST [31], FISTA [29], and NESTA [17], [18]. The strategy of these algorithms is to use smartly-chosen descent directions, that are combinations of the previous two iterates' results. This strategy can be generalized with the sequential subspace optimization (SESOP) [32], [33] for further accelerations, e.g. PCD-SESOP [11], [33] and PCD-SESOP-MM [34]. Variable splitting technique [35], [36] (also known as separable surrogate functionals (SSF) [33]) provides yet another powerful tool to minimize functions that consist of summation of two terms which are of different nature, e.g. the $\ell_2+\ell_1$ objective functional in (3). Several fast algorithms were derived by deploying variable splitting strategy [37]–[41]. Another class of algorithms, which is related to *iterated re-weighted least square* (IRLS), has attracted significant amount of attention, e.g. iterative re-weighted shrinkage [28], which exhibits much faster convergence rate compared with IST.

C. Proposed approach

In this paper, we present a general framework to solve the optimization problem (4) iteratively with a *Linear Expansion of Thresholds* (LET). Our previous work in image denoising showed that it is possible to have a good approximation of the "optimal" denoising process by performing several elementary thresholding processes—which do not need to be chosen particularly carefully—and summing them up with optimized weights [42], [43]. This approximation point of view may be judged similar to a subspace approach [11], [32]–[34], with the philosophical difference that the representation basis may be chosen quite arbitrarily: it is the number of elements that ensure the approximation quality.

The same strategy can be applied to solve the minimization problem (4) here—our idea amounts to approximating the restoration results that minimize (3) with a linear combination of elementary thresholding functions (LET bases) weighted by unknown coefficients (LET coefficients). The original optimization problem, thus, reduces to finding the optimal weights such that the objective functional is minimized. Because of the linearity of the LET expansion, this boils down to solving an ℓ_1-regularized minimization problem *but* with a much smaller dimension. The problem size is determined by the total number of LET bases used. Hence, many conventional convex optimization tools are at our disposal to handle such a small scale problem.

Once we have the optimal linear coefficients, then the reconstruction may be re-synthesized and updated accordingly. We prove that with the elementary thresholding functions satisfying certain properties, convergence to the minimum of (3) is *always* guaranteed. Experimentally, the proposed iterative LET restoration framework is efficient in solving image reconstruction problems and requires much less iterations before it reaches convergence compared with the state of the art algorithms, such as, FISTA [29], SALSA [40], and PCD-SESOP [11] over a wide range of convolution kernels and noise levels. Although each iteration may be more costly than other algorithms, the final computation time is usually still significantly smaller.

D. Organization of the paper

The paper is organized as follows. The basic ingredients of our proposed iterative LET (*i*-LET) restoration method are described in Section II: the *iterative reweighted least square* (IRLS) algorithm and the *linear expansion of thresholds* (LET) approach. The basic LET framework is then adapted to allow iterations in Section III with the convergence proof and several examples of LET bases. Experimental *i*-LET results applied to solving image deconvolution problems are discussed in Section IV before we conclude the paper in Section V.

II. Basic Ingredients

A. Iterative Reweighted Least Square Methods

1) Majorization Minimization point of view: The iterative reweighted least square (IRLS) algorithm can be derived from the *Majorization Minimization (MM)* framework. The MM approach for solving

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} f(\mathbf{x})$$

has the following general form:

$$\mathbf{x}^{(n+1)} = \arg\min_{\mathbf{x}} g\big(\mathbf{x}, \mathbf{x}^{(n)}\big),$$

where $g\big(\mathbf{x}, \mathbf{x}^{(n)}\big)$ "majorizes" $f(\mathbf{x})$ in such a way that $g\big(\mathbf{x}, \mathbf{x}^{(n)}\big) \geq f(\mathbf{x})\ \forall \mathbf{x}$ and $g\big(\mathbf{x}^{(n)}, \mathbf{x}^{(n)}\big) = f\big(\mathbf{x}^{(n)}\big)$. The underlying motivation is that the minimization of $f(\mathbf{x})$ may be very difficult but we can easily minimize $g(\mathbf{x})$, which is an upper bound of $f(\mathbf{x})$. From the majorization definition, it can be easily proven that MM algorithm always produces better estimates when the iterations increase [28], [44]: $f\big(\mathbf{x}^{(n+1)}\big) \leq f\big(\mathbf{x}^{(n)}\big)$.

Unlike IST, which majorizes the Hessian of the quadratic term $\|\mathbf{y} - \mathbf{HWc}\|_2^2$ in the objective functional (3), IRLS majorizes the ℓ_1 penalty instead. Consider $\frac{1}{2}|\tilde{c}| + \frac{1}{2|\tilde{c}|}c^2$, which is an arithmetic average of $|\tilde{c}|$ and $\frac{1}{|\tilde{c}|}c^2$; it is lower bounded by the geometric average $|c|$; hence, $\frac{1}{2|\tilde{c}|}c^2$ is a majorizer of $|c|$ (up to a constant term independent of c). Likewise, a majorization function for the ℓ_1 regularizer $\|\mathbf{c}\|_1$ is $\frac{1}{2}\mathbf{c}^{\mathrm{T}}\mathbf{D}\mathbf{c}$, where $\mathbf{D}$ is a diagonal matrix and $D_{i,i} = \frac{1}{|c_i|}$. Then at the minimization step, $\mathbf{c}$ is updated by minimizing the majorizing function:

$$\mathbf{c}^{(n+1)} = \arg\min_{\mathbf{c}\in\mathbb{R}^D} \Big\{\|\mathbf{y} - \mathbf{HWc}\|_2^2 + \frac{\lambda}{2}\mathbf{c}^{\mathrm{T}}\mathbf{D}^{(n)}\mathbf{c}\Big\},$$

which has the form of a (reweighted) ℓ_2 regularization. The updated iterate $\mathbf{c}^{(n+1)}$ is given by:

$$\mathbf{c}^{(n+1)} = \Big((\mathbf{HW})^{\mathrm{T}}(\mathbf{HW}) + \frac{\lambda}{2}\mathbf{D}^{(n)}\Big)^{-1}(\mathbf{HW})^{\mathrm{T}}\mathbf{y}. \tag{6}$$

IRLS-type algorithms exhibit superior performance in terms of convergence speed among various experiments [6], [28], [45]. Global convergence of IRLS has been shown in [46], [47]. Note that we will not apply IRLS to solve the *straight* ℓ_1 minimization problem (4) (contrary to [28], [48]) but to solve a much smaller dimensional optimization problem (see Section II-B2).

2) Image size issues: It is generally difficult to apply IRLS *directly* to images due to limited computational resources, e.g. memory size and RAM access speed. The computational difficulties arise from the matrix inversion in (6), which requires explicit computation and storage of $\mathbf{HW}$. For instance, the dimension of the inverse matrix $\big((\mathbf{HW})^{\mathrm{T}}(\mathbf{HW}) + \frac{\lambda}{2}\mathbf{D}^{(n)}\big)^{-1}$ is 65536×65536 for a typical 256×256 image when decimated wavelet transform is used. One exception is image denoising, i.e. $\mathbf{H} = \mathbf{I}$, where the inverse matrix becomes diagonal and the matrix inversion can be computed component-wise (but the exact solution is already known: component-wise soft-threshold). Otherwise, without dimension reduction, the linear system (6) has to be solved iteratively. In [28], [31], [49], *second-order stationary iterative method* (SOSIM) is proposed to solve the system with a few inner iterations for each MM iteration.

B. Linear expansion of thresholds (LET)

1) LET expansion: From our previous experience in image denoising, we know that it is possible to have an excellent estimate of the noiseless image by decomposing the denoising process into a linear combination of elementary thresholding processes—*Linear Expansion of Thresholds* (LET) and then optimizing the coefficients of this representation using an estimate of the MSE—the *Stein's Unbiased Risk Estimate* (SURE) [42], [43]. Here we use the same strategy by representing the solution of (4) as a function of the known data $\mathbf{y}$ from a linear combination of thresholding processes $\mathbf{F}_k(\mathbf{y})$:

$$\mathbf{c} = \sum_{k=1}^{K} a_k \mathbf{F}_k(\mathbf{y}), \tag{7}$$

where a_k's for $k = 1, \ldots, K$ are linear coefficients to be determined. Note that even though here we term (7) as a linear expansion of "thresholds" for consistency with our previous work in denoising, the actual meaning is "elementary processings"; i.e., "basis" vector-functions that transform $\mathbf{y}$ into partially-restored wavelet coefficients. Such bases may be actual (non-linear) thresholds (see Section II-B4), Tikhonov linear estimates (see Section II-B3), or recursive estimates (functions of previous estimates, see Section III).

Therefore, instead of directly addressing the *solution* that minimizes (3), our goal is to approximate the optimal *function* that maps the distorted measurements to the reconstruction, which makes this approach akin to approximation theory and Ritz-Galerkin method [50]. Specifically, for a given set of elementary functions $\mathbf{F}_k$, the output of which gives a reconstruction candidate, we only need to determine the optimal linear combinations $a_1, \ldots, a_K$. Here the optimality is in the sense that the objective functional (3) is minimized:

$$a_k = \arg\min_{a_k\in\mathbb{R}} J\bigg(\sum_{k=1}^{K} a_k \mathbf{F}_k(\mathbf{y})\bigg). \tag{8}$$

Thus the algorithm automatically takes the best of each LET basis $\mathbf{F}_k$ via the optimization of LET coefficients. It is worth mentioning that even though we parametrize the reconstruction process with a *linear* combination of LET basis elements, the resultant function is usually not a linear mapping between the measurements $\mathbf{y}$ and the reconstruction $\hat{\mathbf{x}}$. Indeed, by choosing adequate and sufficiently many elementary thresholding functions $\mathbf{F}_k$, a rich variety of functions can be constructed.

2) Solving for LET coefficients: If we introduce the parametric LET representation (7) into (3), then the resultant objective functional is still convex in a_k's, with the advantage of a much smaller dimensionality: the problem's dimension changes from image size ($\approx 6.5\times10^4$ for a typical 256×256 image) to the degrees of freedom of LET basis K (typically less than 10). Therefore, the minimization problem can be easily addressed with standard convex optimization tools, e.g. *iterated reweighted least square* approach that we introduced previously in Section II-A (also see [34] for a similar strategy).

More specifically, when IRLS is used, the LET coefficients are obtained *iteratively* as:

$$\mathbf{a}^{(n+1)} = \arg\min_{\mathbf{a}\in\mathbb{R}^K} \left\{ \left\|\mathbf{y}-\mathbf{HWFa}\right\|_2^2 + \frac{\lambda}{2}(\mathbf{Fa})^{\mathrm{T}}\mathbf{D}^{(n)}\mathbf{Fa} \right\},$$

where we have rearranged the LET coefficients as a vector $\mathbf{a} = [\ a_1 \ \ a_2 \ \ \ldots \ \ a_K\ ]^{\mathrm{T}}$, and the LET basis elements in a matrix form $\mathbf{F} = [\ \mathbf{F}_1 \ \ \mathbf{F}_2 \ \ \ldots \ \ \mathbf{F}_K\ ]$. Here $\mathbf{D}^{(n)}$ is a diagonal matrix such that[1] $\left[\mathbf{D}^{(n)}\right]_{i,i} = 1/\left[\left|\mathbf{Fa}^{(n)}\right|\right]_i$. Then, we only need to solve a small linear system of equations at each iteration:

$$\mathbf{a}^{(n+1)} = \left(\mathbf{M}^{(n)}\right)^{-1}\mathbf{b}, \tag{9}$$

where $\mathbf{b}$ is a $K\times 1$ vector and $\mathbf{M}^{(n)}$ is a $K\times K$ matrix, given by:

$$\begin{aligned} \mathbf{M}^{(n)} &= (\mathbf{HWF})^{\mathrm{T}}(\mathbf{HWF}) + \frac{\lambda}{2}\mathbf{F}^{\mathrm{T}}\mathbf{D}^{(n)}\mathbf{F}, \\ \mathbf{b} &= (\mathbf{HWF})^{\mathrm{T}}\mathbf{y}. \end{aligned} \tag{10}$$

Once we have the LET coefficients $\mathbf{a}^{(\infty)}$, the image is re-synthesized as $\hat{\mathbf{x}} = \mathbf{W}\hat{\mathbf{c}} = \mathbf{WFa}^{(\infty)}$. Note that here we are not limited to the use of IRLS to obtain the LET coefficients, as long as the algorithm can solve (8) efficiently. Experimentally, IRLS reaches "convergence" within a few iterations (typically less than 10).

3) Example of LET restoration—Linear Processing: We use Tikhonov regularizers as LET bases: $\mathbf{F}_k(\mathbf{y}) = \mathbf{W}^{\mathrm{T}}(\mathbf{H}^{\mathrm{T}}\mathbf{H}+\mu_k\mathbf{I})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{y}$ for some μ_k's, where $\mathbf{H}$ is the convolution matrix and $\mathbf{I}$ is the identity matrix. Here we only need to coarsely select several μ_k's such that the LET bases reflect different aspects of the reconstruction. Fig. 1 shows the result of three LET processings $\mathbf{F}_k$ and their J-optimal combination. In comparison, if we hand-tune the parameter μ, then the objective functional J in (3) is worse than what we obtained with the LET approach. Therefore, we can out-perform an empirically selected *non-linear* parameter with several *linear* parameters (i.e. the LET coefficients). The computational advantage is obvious, as well as the certainty to have a global minimum (because of convexity).

4) Example of LET restoration—Linear Processing + Thresholding: We can decrease the objective functional substantially by further (non-linear) thresholding the wavelet coefficients of the Tikhonov deconvolution, which results in a new set of LET basis elements. A possible choice of the non-linear part of this processing is a combination of the ramp and of a derivative of Gaussian (DOG) as defined in [42]: $\theta_{\mathrm{DOG}}(c;\mathbf{a}) = \left(a_1 + a_2 \mathrm{e}^{-\frac{c^2}{12\sigma^2}}\right)c$, where $\mathbf{a} = [a_1 \ \ a_2]^{\mathrm{T}}$ are (LET) coefficients to be determined by minimizing the objective functional (3), σ is the standard deviation of the additive noise, which are either given or can be robustly estimated [51], and c corresponds to each component of the wavelet coefficients of the Tikhonov deconvolution results $\mathbf{W}^{\mathrm{T}}(\mathbf{H}^{\mathrm{T}}\mathbf{H}+\mu_k\mathbf{I})^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{y}$. This one-step LET reconstruction is already reasonably close (in terms of peak signal to noise ratio—see the definition (16)) to the exact minimizer of the objective functional (3) with various experimental settings.

[1]To avoid numerical instability, we use $\max\left(|\mathbf{Fa}^{(n)}|_i, 10^{-15}\right)$ for the denominator in $\mathbf{D}^{(n)}$ in the implementation.

TABLE II
PSNR COMPARISON BETWEEN THE NON-ITERATIVE LET RECONSTRUCTION OF SECTION II-B4 AND THE EXACT SOLUTION OF (4)—REDUNDANT WAVELET TRANSFORM (SYM8, THREE STAGES)

BSNR	25	30	35	40
Blur Type	***cameraman*** (256×256)			
1	28.83	29.06	29.74	30.70
2	32.57	32.95	35.19	37.21
3	34.87	35.18	35.03	34.99

NOTE: The similarity (in dB) between the non-iterative LET reconstruction and the exact solution of the ℓ_1-regularized minimization (4) is measured by the peak signal to noise ratio defined in (16). See Table IV for the definitions of various convolution kernels.

Table I and Table II summarises the results we obtained when orthogonal and redundant wavelet transforms are used for the transformation respectively.

III. ITERATIVE LET RESTORATION

In the previous section we have proposed a general form (7) for the approximation of the restoration function. The algorithm finds the best linear combination of LET basis elements that optimizes the objective functional but it will not find the exact solution of (4), in general. It is possible, however, to apply the LET strategy *iteratively* (*i*-LET) in such a way as to refine the solution provided by previous iterations (Fig. 2): at each iteration, it suffices to build LET bases that are functions of both the measurements $\mathbf{y}$ and the previous iterate results $\mathbf{c}^{(n)}$: $\mathbf{F}\big(\mathbf{y},\mathbf{c}^{(n)}\big)$. And the optimal weighting coefficients are given by

$$a_k = \arg\min_{a_k\in\mathbb{R}} J\left(\sum_{k=1}^{K} a_k \mathbf{F}_k(\mathbf{y},\mathbf{c}^{(n)})\right). \tag{11}$$

This choice makes our algorithm related to standard optimization approaches like (conjugate) gradient-descent, coordinate-descent [52] and SESOP [33], although our specificity here rests on a non-differentiable optimization criterion. We prove that, under very mild conditions on the choice of LET bases, this iterative procedure eventually results in the minimum of (3).

Algorithm 1: Iterative LET Restoration

Input : Distorted measurements $\mathbf{y}$, distortion matrix $\mathbf{H}$
Output: Reconstructed image $\hat{\mathbf{x}}$

while *stopping criterion not met* **do**
1 Build LET bases from measurements $\mathbf{y}$ and previous iterate results $\mathbf{c}^{(n)}$;
2 Solve LET coefficients $\mathbf{a}$ from $a_k = \arg\min_{a_k} J\Big(\sum_{k=1}^{K} a_k\mathbf{F}_k\big(\mathbf{y},\mathbf{c}^{(n)}\big)\Big)$;
3 Re-synthesize reconstruction $\mathbf{c}^{(n+1)} = \sum_{k=1}^{K} a_k\mathbf{F}_k\big(\mathbf{y},\mathbf{c}^{(n)}\big)$.
end
Reconstruct solution $\hat{\mathbf{x}} = \mathbf{W}\mathbf{c}^{(\infty)}$.

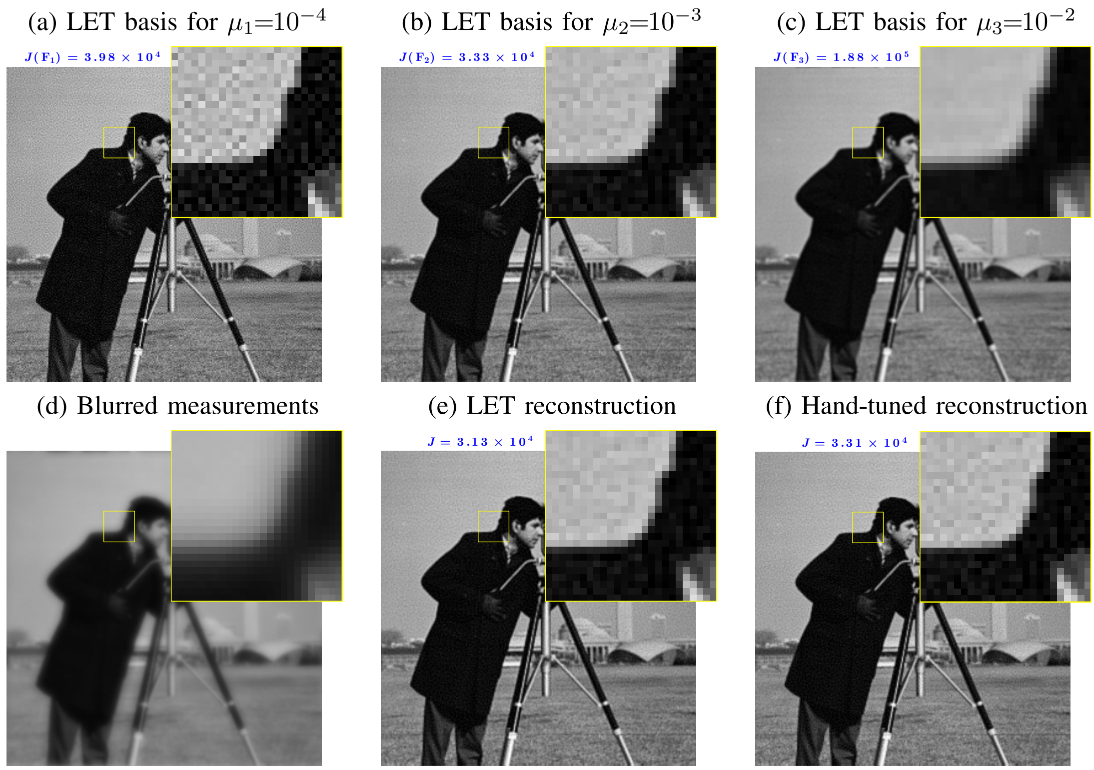


Fig. 1. (a)–(c) LET bases built from Tikhonov deconvolution with different regularization weights μ_k's, (d) the blurred *cameraman* image (e) the LET reconstruction, i.e. the J-optimal combination of (a), (b) and (c) (J given in (3)), and (f) the Tikhonov deconvolution with hand-tuned parameter. (see, text in Section II-B3)

TABLE I
PSNR COMPARISON BETWEEN THE NON-ITERATIVE LET RECONSTRUCTION OF SECTION II-B4 AND THE EXACT SOLUTION OF (4)—ORTHOGONAL WAVELET TRANSFORM (SYM8, THREE STAGES)

BSNR	10	15	20	25	30	35	40	10	15	20	25	30	35	40
Blur Type	***cameraman*** (256 × 256)							***bank*** (512 × 512)						
1	30.39	29.02	27.93	28.35	28.09	28.51	29.42	29.07	28.58	27.68	27.81	27.86	28.75	30.10
2	31.34	30.14	31.46	33.18	33.48	34.50	36.96	31.36	30.74	32.08	34.14	35.94	35.80	38.37
3	30.58	32.45	34.51	33.27	31.45	30.94	31.18	31.06	33.16	35.51	37.44	34.09	33.00	32.84
Blur Type	***peppers*** (256 × 256)							***mandrill*** (512 × 512)						
1	30.00	28.30	27.73	26.98	27.41	28.69	30.38	32.16	29.40	28.37	27.89	28.23	28.87	29.94
2	32.23	31.67	33.06	35.12	36.42	36.31	38.46	31.58	32.40	33.48	32.76	33.44	35.89	38.74
3	32.01	34.07	36.67	39.85	35.81	34.37	34.14	32.98	34.67	33.10	30.27	29.29	29.14	29.30

NOTE: The similarity (in dB) between the non-iterative LET reconstruction and the exact solution of the ℓ_1-regularized minimization (4) is measured by the peak signal to noise ratio defined in (16). See Table IV for the definitions of various convolution kernels.

A. *Selection of i-LET bases*

In principle, we have the freedom to choose arbitrary elementary LET processings. However, in an iterative framework it is reasonable to describe the update as a sum of the previous iterate and an incremental change. Hence, a typical minimal LET basis is made of two elements: the previous iterate, and a change that we are now going to specify.

Which constraint should we impose on the incremental change so as to ensure convergence of the iterations to the global minimum of (3)? For continuously-differentiable objective criteria, an instance of converging iterative scheme is the gradient-descent algorithm. Under the *i*-LET framework, this means that the second LET basis element should be the gradient of the objective functional, ∇J itself, as shown in [32]. Despite the fact that J in (3) is not continuously-differentiable, it is still possible to define a LET element that behaves like a gradient; we denote by $\overline{\nabla}_\tau J$ this "generalized" gradient:

$$\overline{\nabla}_\tau J \stackrel{\text{def}}{=} \frac{2}{\tau}\left(\mathbf{c} - \theta_{\lambda\tau/2}\left(\mathbf{c} - \frac{\tau}{2}\nabla J_0\right)\right), \tag{12}$$

where τ is an arbitrary positive number and $J_0(\mathbf{c}) = \|\mathbf{y} - \mathbf{HWc}\|_2^2$. We will see that a small variation of $\mathbf{c}$ opposite to this "gradient" results in a decrease of $J(\mathbf{c})$, unless $\mathbf{c}$ minimizes $J(\mathbf{c})$, in which case $\overline{\nabla}_\tau J(\mathbf{c}) = \mathbf{0}$. Any LET basis that contains the previous iterate and $\overline{\nabla}_\tau J$ will be shown to converge to the global minimum of (3). Adding other basis may speed up the algorithm, but does not alter the convergence behavior. These requirements are summarized in Table III.

B. *Convergence of the* i-*LET scheme*

We first give an equivalent characterization of the minimum of (3), and a sufficient condition for this minimum to be unique, as they appear in the literature. Then, we prove the

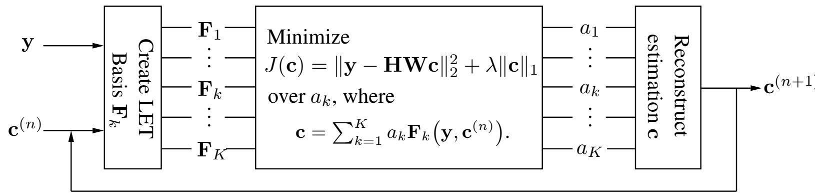


Fig. 2. *i*-LET deconvolution schematic view. For a given set of LET bases, LET coefficients a_k's are obtained by minimizing (3). The reconstructions are fed back as the input for the next iterate.

TABLE III
ITERATIVE LET BASES SELECTION CRITERIA

A "good" set of thresholding bases should contain:
1 ***Previous iterate***: at each iteration the updated result obtained from the algorithm is no worse than the one at the previous iteration;
2 ***Generalized gradient***: $\overline{\nabla}_\tau J \stackrel{\mathrm{def}}{=} \frac{2}{\tau}\big(\mathbf{c} - \theta_{\lambda\tau/2}(\mathbf{c} - \frac{\tau}{2}\nabla J_0)\big)$ for arbitrary $\tau > 0$ where $J_0 = \|\mathbf{y} - \mathbf{HWc}\|_2^2$ is the quadratic term in the objective functional J given in (3).

unconditional convergence of the value of the criterion (3) to its *global* minimum in the iterative LET scheme. When the problem is known to have a unique solution, we will finally show that the LET iterates converge to this very minimum.

Lemma 1 (adapted from [21, Proposition 3.1 (iii)]): $\mathbf{c}^\star \in \mathbb{R}^D$ minimizes the criterion (3), if and only if there exists $\tau > 0$ such that $\mathbf{c}^\star$ satisfies

$$\overline{\nabla}_\tau J(\mathbf{c}^\star) = \mathbf{0}. \tag{13}$$

Note that, when (13) is satisfied for some $\tau > 0$ then it will be satisfied for *any* $\tau > 0$. The following theorem provides a sufficient condition that we will explicitely check in the experiment Section IV.

Theorem 1 (from [53]–[55]): Let $\mathbf{c}^\star \in \mathbb{R}^D$ be a minimizer of (3) and define $S(\mathbf{c}^\star) = \{i \in \{1, \dots, D\} : \mathbf{c}_i^\star \neq 0\}$, then the solution $\mathbf{c}^\star$ is unique if

$$\begin{aligned} \mathbf{A}_i^{\mathrm{T}}(\mathbf{y} - \mathbf{Ac}^\star) &= \frac{\lambda}{2}\mathrm{sign}(\mathbf{c}_i^\star), \quad \text{for } i \in S(\mathbf{c}^\star),\\ \big|\mathbf{A}_i^{\mathrm{T}}(\mathbf{y} - \mathbf{Ac}^\star)\big| &< \frac{\lambda}{2}, \quad \text{for } i \notin S(\mathbf{c}^\star),\\ \mathbf{A}_{S(\mathbf{c}^\star)}^{\mathrm{T}}\mathbf{A}_{S(\mathbf{c}^\star)} &\text{ is invertible.} \end{aligned}$$

Here $\mathbf{A} = \mathbf{HW}$ and $\mathbf{A}_{S(\mathbf{c}^\star)}$ extracts columns in $\mathbf{A}$ that are indexed by $S(\mathbf{c}^\star)$.

We now state a strictly decreasing property satisfied by $\overline{\nabla}_\tau J$, which outlines its similarity with a true gradient (if J were differentiable).

Lemma 2: Let $\overline{\nabla}_\tau J$ be the generalized gradient of (3) where $\tau > 0$ is arbitrary. Then $J(\mathbf{c} - \varepsilon\,\overline{\nabla}_\tau J) < J(\mathbf{c})$ for sufficiently small $\varepsilon > 0$ unless $\overline{\nabla}_\tau J = \mathbf{0}$.

Proof: See Appendix A-A. ■

What is important here is the *strict* inequality: in the neighborhood of a (non-optimal) point $\mathbf{c}$, descending in the direction of $-\overline{\nabla}_\tau J(\mathbf{c})$ decreases the value of the criterion *strictly*. Note however that, contrary to the true gradient (of a continuously differentiable function), preconditioning the generalized gradient does not automatically decrease the criterion, even for values of $\mathbf{c}$ away from convergence. Said differently, $-\mathbf{Q}\,\overline{\nabla}_\tau J$ is not always a descent direction of J, even if $\mathbf{Q} \succ \mathbf{0}$.

Lemma 2 is instrumental in proving the convergence of iterated LET schemes for arbitrary values of $\tau > 0$ in the following theorem.

Theorem 2: Let $\tau > 0$, then the iterated LET restoration (Algorithm 1) minimizes the criterion (3). More precisely, any limit points of the *i*-LET sequence minimize this criterion.

If, in addition, we are ensured of the unicity of the minimum, then the iterated estimates $\mathbf{c}^{(n)}$ converge towards this minimum.

Proof: See Appendix A-B. ■

How does this simple result stand compared to other convergence results in the literature?

- SESOP [32] uses a smooth approximation of the ℓ_1 norm and is not able to solve exactly the non-differentiable ℓ_1 problem: when the approximation of ℓ_1 gets more rough, convergence slows down substantially because the true gradient of the criterion becomes more unstable. What we have proven here is that this true gradient has to be replaced by a generalized gradient (12). With this modification, the *i*-LET iterations converge to the rough *non-differentiable* ℓ_1 problem. Moreover, even if at each iteration the LET coefficients are not optimized exactly according to the ℓ_1 criterion (due to, e.g. few IRLS iterations, 10^{-15} saturation), convergence to the solution of the ℓ_1 problem is still very likely[2] to hold.
- IST-like schemes are known to be convergent when τ is smaller than the spectral radius of the distortion matrix. The obvious improvement here is that τ may be arbitrary. On the other hand, our result addresses less general situations than, e.g., [4], [21] because of our finite-dimension setting, which ensures that *i*-LET sequences have limit points.

C. Examples of i-LET bases

In this section, we exemplify the iterative algorithm with different choices of LET bases, which satisfy the properties summarized in Table III. Therefore, from Section III-B, we know that global convergence is guaranteed. However, it is expected that with different sets of bases, the convergence speed varies. Therefore, the difficulty now lies on designing adequate LET bases. For instance, we could have chosen the

[2]This is because the IST, which is itself an *i*-LET algorithm with fixed coefficients, is known to converge to the solution of the exact non-smooth problem (4). Hence, the choice of the coefficients (even through ill-adapted optimization) is *not* expected to lead to a different solution.

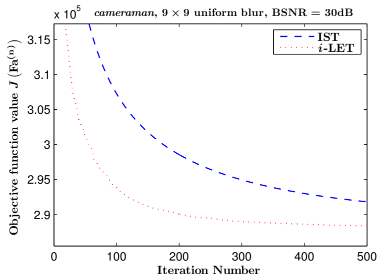


Fig. 3. Convergence comparison between IST and *i*-LET (with same $\tau = 0.5$) where the LET basis consists of previous iterate result and a soft-threshold—see (14).

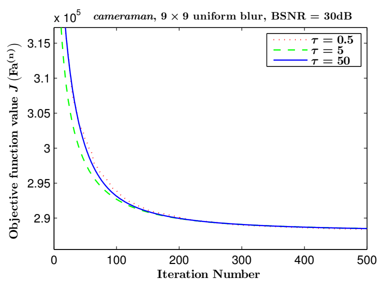


Fig. 4. The *i*-LET algorithm converges to the global optimum for arbitrary values of τ (IST would diverge for $\tau \geq 1$).

thresholding bases introduced in Section II-B4, and added the ones recommended in Table III. But our experiments indicated that the computational cost caused by having many LET bases ($57 + 2$ here) could not be counterbalanced by having less iterations. For this reason, we chose to design fewer LET bases.

We first focus on one particular LET basis that follows naturally from the IST algorithm. It is also the simplest set of thresholding functions that conform to the convergence requirements, and serves as a reference for discussions of other alternative bases that further improve the convergence rate of *i*-LET.

1) IST-like bases: The LET bases consist of the previous iterate $\mathbf{c}^{(n)}$ and the generalized gradient (12):

$$\mathbf{F}_1 = \mathbf{c}^{(n)}, \quad \text{and} \quad \mathbf{F}_2 = \overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big). \tag{14}$$

The IST is actually a LET with *fixed* weights throughout iterations: $a_1 = 1$ and $a_2 = -\tau/2$ for $\mathbf{F}_1$ and $\mathbf{F}_2$ respectively. In contrast, the *i*-LET algorithm consists in optimizing these two weights at each iteration according to (11). Thanks to this optimization, notable accelerations are readily observed in experiments compared with the standard IST (Fig. 3).

It is important to stress again that, contrary to the IST, we do not have here any further limitations on the positive step-size τ of the *i*-LET for its convergence to the global minimum. We exemplify this insensitivity to τ in Fig. 4.

2) Bases with more than one previous iterate: It is well known that the convergence rate is dramatically improved by simply taking into account previous two or more iterate results [29], [31], [49]. Nesterov has proven that with a smartly chosen additional iterate result to the previous one, *subquadratic* convergence rate can be achieved [17], [18].

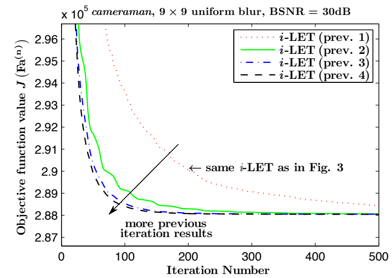


Fig. 5. Comparison of *i*-LET convergence with different number of previous iterates results included as LET bases—see Section III-C2. (Note: narrowing of vertical axis compared to Fig. 3)

Such remarkable results give us the motivation to derive the following LET bases:

$$i\text{-LET prev. 2:} \quad \begin{cases} \mathbf{F}_1 = \mathbf{c}^{(n-1)} \\ \mathbf{F}_2 = \mathbf{c}^{(n)} \\ \mathbf{F}_3 = \overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big) \end{cases},$$

$$i\text{-LET prev. 3:} \quad \begin{cases} \mathbf{F}_1 = \mathbf{c}^{(n-2)} \\ \mathbf{F}_2 = \mathbf{c}^{(n-1)} \\ \mathbf{F}_3 = \mathbf{c}^{(n)} \\ \mathbf{F}_4 = \overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big) \end{cases}, \quad \text{etc.}$$

In this sense, the LET basis considered in (14) is a special case of the ones presented here. Fig. 5 are the results obtained with *i*-LET when different numbers of previous iterates are involved in the LET bases. We found that the convergence rate is insensitive to the number of previous iterates used after a certain point (here approximately three or even two seems to be enough). Similar behavior was observed in [49]. In practice, we would like to use as few LET basis elements as possible to minimize the computation cost of the *i*-LET algorithm (see below in Section IV-C). Previous efforts to build more complicated shrinkage functions other than the soft-threshold by combining several thresholding functions [31], [49], experience difficulties in specifying proper weight for each individual thresholding function. Empirical approaches were usually adopted to optimize the convergence speed. With the LET approach, however, it becomes almost trivial to determine the proper weights, as the minimization of the objective functional automatically takes care of the necessary adjustments for fastest decrease.

3) Bases with preconditioned generalized gradient: The LET bases discussed in the previous Sections III-C1 and III-C2 result in *i*-LET algorithms that are akin to gradient-descent algorithms. Therefore, only first order information is considered. Faster convergence is expected when second order information of the objective functional is taken into account, e.g. the Hessian, especially when the problem is very ill-conditioned. In particular, in analogy with preconditioned gradient algorithms, we consider the following LET basis:

$$\begin{aligned} &\mathbf{F}_1 = \mathbf{c}^{(n)}, \\ &\mathbf{F}_2 = \overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big), \\ &\mathbf{F}_3 = \big(\mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{W} + \mu\mathbf{I}\big)^{-1}\,\overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big). \end{aligned} \tag{15}$$

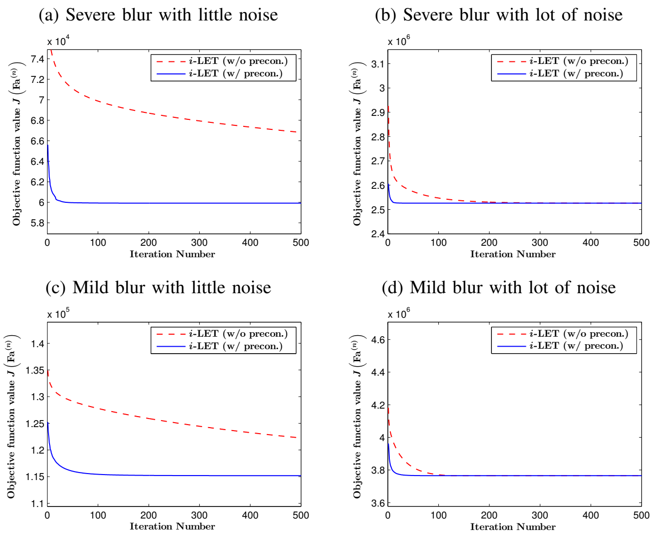


Fig. 6. Convergence speed improvements by including preconditioned generalized gradient (15) in addition to the previous iterate and the soft-threshold basis under different levels of blur and/or noise (image: *cameraman*).

TABLE IV
CONVOLUTION KERNELS USED IN EXPERIMENTS

Convolution Type	Description
Type 1	9×9 uniform blur
Type 2	$h_{i,j} = 1/(1+i^2+j^2)$ for $i,j = -7,\ldots,7$
Type 3	$h_{i,j} = [1,4,6,4,1]^{\mathrm{T}}[1,4,6,4,1]/256$

Here the regularized Hessian $\left(\mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{W} + \mu\mathbf{I}\right)^{-1}$ is used for "preconditioning" the generalized gradient.[3] Now comes an obvious question: how to select the optimal μ, which is case-dependent, for fast convergence? We can either choose it empirically *or* we can do it "the LET way": use several bases of the same form as $\mathbf{F}_3$ but with different μ's and let the algorithm take necessary adjustments by assigning different weights to each basis element. The convergence rate improvements under various blurring and/or noise level conditions are shown in Fig. 6. The speedup is more obvious when the measurements $\mathbf{y}$ are less noisy.

IV. EXPERIMENTAL RESULTS

In this section, we exemplify the proposed iterative LET restoration algorithm to solve image deconvolution problems where the images have been distorted with various convolution kernels (Table IV) and different levels of noise. Both decimated wavelet transform (DWT) and redundant wavelet transform (RWT) have been used. We compare the performances of the proposed *i*-LET algorithm with the state of art algorithms that are available in the public domain[4]: FISTA [29], SALSA [40] and PCD with subspace accelerations (PCD-SESOP) [11], [33]. We have run all experiments with Matlab 7.9.1 in Windows 7 (64-bit) on a desktop computer with Intel Core i3 3.1GHz CPU and 4GB RAM. We did not make use of any additional pre-compiled code, but special treatments had to be made in order to be more efficient in building the IRLS matrix in (10) due to Matlab's memory management behavior.

In the experiments, the parameters of FISTA, SALSA and PCD-SESOP have been set to their default values as suggested by their corresponding authors. We have initialized all comparing algorithms with the same random wavelet coefficients, and averaged the computation times and iteration numbers over ten realizations of this random initialization.

In order for our stopping criterion to be as objective as possible, we first obtain the global minimizer $\mathbf{c}^\star$ of the ℓ_1-regularized objective functional (3) before we run all four algorithms for comparisons. This global minimizer is computed by running FISTA with usually more than 10^6 iterations and checking that we are effectively at convergence using Lemma 1. We also explicitly verify that this solution $\mathbf{c}^\star$ is *unique* by checking that the conditions in Theorem 1 are met for all cases considered in the subsequent sections.[5]

We terminate the iterations when the reconstructed image $\hat{\mathbf{x}} = \mathbf{W}\hat{\mathbf{c}}$ from each algorithm is close enough to the one at convergence $\mathbf{x}^\star = \mathbf{W}\mathbf{c}^\star$ in peak signal to noise ratio (PSNR) sense:

$$\mathrm{dist}(\mathbf{x},\mathbf{x}^\star) \stackrel{\mathrm{def}}{=} 10\log_{10}\left(\frac{255^2}{\mathrm{MSE}(\mathbf{x},\mathbf{x}^\star)}\right) \geq 40\,\mathrm{dB}. \qquad (16)$$

We observed that this stopping criterion corresponds to a relative difference of 10^{-4} up to 10^{-3} (depending on the experimental settings) between the resulting objective function and the minimum $J(\mathbf{c}^\star)$.

A. Deconvolution with decimated wavelet transform

We consider image deconvolution with a decimated wavelet transform over a wide range of noise variations (σ^2). In particular, the blurred signal to noise ratio (BSNR)

$$\mathrm{BSNR}_{[\mathrm{dB}]} \stackrel{\mathrm{def}}{=} 10\log_{10}\left(\frac{\mathrm{Var}(\mathbf{Hx})}{\mathrm{Var}(\mathrm{noise})}\right)$$

varies from $10\,\mathrm{dB}$ to $40\,\mathrm{dB}$. And we used three different convolution kernels as summarized in Table IV. In all experiments, `sym8` decimated wavelet transform with three stages is used. The regularization weight λ is chosen in such a way that the data-fidelity term is compatible with the noise level at convergence $\|\mathbf{y} - \mathbf{HW}\hat{\mathbf{c}}\|_2^2 = N\sigma^2$ for an N-pixel image (see Section IV-D). We only used the wavelet subbands in the ℓ_1 regularization (not the low-pass), since the low-pass subband of a natural image is generally not sparse. Four different test images are used in comparisons, namely *cameraman* (256×256), *bank* (512×512), *peppers* (256×256) and *mandrill* (512×512). We use the following set of LET

[3] Note that this "preconditioned" generalized gradient is *not* always a descent direction for J (see our comments below Lemma 2), which is why we need to keep $\overline{\nabla}_\tau J$.

[4] FISTA: http://iew3.technion.ac.il/~becka/papers/wavelet_FISTA.zip
SALSA: http://cascais.lx.it.pt/~mafonso/SALSA_v1.0.zip
PCD-SESOP: http://ie.technion.ac.il/~mcib/SESOP_PACK_07_06_2010.zip

[5] We use LU decomposition of $\mathbf{A}^{\mathrm{T}}_{S(\mathbf{c}^\star)}\mathbf{A}_{S(\mathbf{c}^\star)}$ to check the invertibility, which usually takes a few hours for each case. The minimum absolute value in the diagonal entries of the upper-triangular matrix is around 10^{-4} for DWT cases and 10^{-6} for RWT cases.

TABLE V
DECONVOLUTION CONVERGENCE COMPARISONS WITH ORTHOGONAL WAVELET TRANSFORM (SYM8, THREE STAGES)

BSNR		10	15	20	25	30	35	40	10	15	20	25	30	35	40
Method		***cameraman*** (256×256) type **1** blur							***bank*** (512×512) type **1** blur						
FISTA	iterations	18	33	64.1	72	112	147	176	21	31	55	89	130	162	176
	computation time	0.30	0.49	0.92	1.02	1.58	2.08	2.45	1.67	2.37	4.16	6.68	9.72	12.55	14.23
SALSA	iterations	60	58	75	40	40	31	20	72	46	51	55	52	38	22
	computation time	1.26	1.28	1.57	0.86	0.9	0.66	0.46	7.65	4.78	5.31	5.7	5.44	3.99	2.39
PCD-SESOP-7	iterations	234.1*	151.7	46.1	34.6	43.4	56.3	72.4	741.2*	84.8	44	39.1	48.6	61.3	76.5
	computation time	10.82	7.19	2.19	1.64	2.05	2.64	3.39	165.92	18.97	9.86	8.75	10.88	13.66	17.09
i-LET	iterations	**6**	**7**	**10**	**8**	**8.9**	**9.2**	**9.9**	**7**	**6**	**8**	**10**	**10.4**	**10.1**	**10.5**
	computation time	**0.28**	**0.29**	**0.40**	**0.32**	**0.35**	**0.36**	**0.39**	**1.42**	**1.23**	**1.57**	**1.93**	**2.03**	**1.98**	**2.04**
Method		***cameraman*** (256×256) type **2** blur							***bank*** (512×512) type **2** blur						
FISTA	iterations	9	14	18	22	36	52	63	9	13	16	19	25	41	53
	computation time	0.16	0.22	0.27	0.33	0.53	0.74	0.90	0.74	1.05	1.26	1.49	1.93	3.37	4.32
SALSA	iterations	19	15	11	7	7	6	4	18	13	9	6	4	**4**	**3**
	computation time	0.42	0.34	0.27	0.18	0.18	**0.16**	**0.12**	1.99	1.50	1.09	0.78	**0.60**	**0.59**	**0.48**
PCD-SESOP-7	iterations	238	74.1	36.2	16.6	10	10.2	11	219.7	72.8	36.5	16.7	10	10.1	10.9
	computation time	11.10	3.53	1.73	0.80	0.48	0.49	0.53	48.85	16.30	8.20	3.78	2.26	2.29	2.45
i-LET	iterations	**3**	**3.5**	**3**	**3**	**4**	**4**	**3.4**	**3**	**3**	**3**	**3**	**3**	**4**	**3**
	computation time	**0.14**	**0.16**	**0.14**	**0.13**	**0.17**	0.17	0.15	**0.65**	**0.65**	**0.65**	**0.65**	0.66	0.85	0.66
Method		***cameraman*** (256×256) type **3** blur							***bank*** (512×512) type **3** blur						
FISTA	iterations	5	6	7	52	107.9	172.9	249.8	4	5	6	8	62	116	166
	computation time	**0.09**	**0.10**	**0.11**	0.74	1.5	2.43	3.48	**0.37**	**0.45**	0.51	0.66	4.66	9.39	13.42
SALSA	iterations	21	9	5	54	93	109	107	18	8	4	**3**	34	52	52
	computation time	0.48	0.23	0.14	1.13	1.98	2.32	2.30	2.02	0.98	0.58	**0.48**	3.60	5.38	5.45
PCD-SESOP-7	iterations	116.4	77.9	41.4	26.9	35.4	62	90.7	114	71.7	42	24.8	27.8	54.4	77.6
	computation time	5.54	3.76	1.94	1.27	1.68	2.96	4.28	25.43	16.09	9.42	5.59	6.27	12.31	17.41
i-LET	iterations	**3.2**	**3.1**	**3**	**11**	**16.2**	**18.6**	**19.4**	**3**	**3.1**	**2.1**	3.1	**11.2**	**15.3**	**14.9**
	computation time	0.15	0.15	0.14	**0.44**	**0.61**	**0.71**	**0.74**	0.64	0.67	**0.49**	0.67	**2.21**	**2.94**	**2.86**
Method		***peppers*** (256×256) type **1** blur							***mandrill*** (512×512) type **1** blur						
FISTA	iterations	21	38	83.1	167	241	272.5	270.5	20	41	100	150	169	201	248.7
	computation time	0.31	0.54	1.16	2.32	3.35	3.77	3.77	1.55	3.13	7.53	11.25	12.76	15.07	18.72
SALSA	iterations	43	46	82	152	146	85	38	22	31	67	64	37	23	16
	computation time	0.90	0.94	1.66	3.03	2.96	1.75	0.79	2.39	3.30	6.88	6.63	3.93	2.76	**1.96**
PCD-SESOP-7	iterations	466.5*	55.3	37.2	36.2	46.8	59.9	78.8	62.6	40.5	36.2	47.8	63.9	85.9	117
	computation time	20.77	2.55	1.71	1.67	2.15	2.70	3.62	14.07	9.39	8.51	11.10	14.82	19.23	26.09
i-LET	iterations	**5**	**6.8**	**12**	**19.9**	**19.6**	**15.5**	**13.2**	**4.1**	**6**	**11.7**	**12.7**	**12.3**	**13.4**	**15.5**
	computation time	**0.21**	**0.28**	**0.46**	**0.75**	**0.72**	**0.58**	**0.50**	**0.85**	**1.20**	**2.30**	**2.46**	**2.39**	**2.66**	3.03
Method		***peppers*** (256×256) type **2** blur							***mandrill*** (512×512) type **2** blur						
FISTA	iterations	9	13	17	20	28	51	69	13	17	23	45	63	74	82
	computation time	**0.14**	0.19	0.25	0.29	0.40	0.72	0.97	1.03	1.33	1.77	3.42	4.75	5.63	6.18
SALSA	iterations	13	10	7	5	4	**4**	4	12	9	6	8	7	5	**3**
	computation time	0.29	0.24	0.17	**0.14**	**0.12**	**0.12**	**0.12**	1.38	1.08	0.79	0.98	**0.89**	**0.77**	**0.52**
PCD-SESOP-7	iterations	138.4	58.3	28.3	14	9	10.5	10.9	67.4	33	15.5	10	10.9	11	11.7
	computation time	6.20	2.69	1.31	0.64	0.42	0.49	0.50	18.79	7.72	3.72	2.37	2.46	2.46	2.65
i-LET	iterations	**3**	**3**	**3**	**3**	**3**	**4**	**3**	**3.8**	**3.1**	**3**	**4**	**4.8**	**4**	4
	computation time	**0.14**	**0.14**	**0.14**	**0.14**	0.14	0.17	0.14	**0.81**	**0.66**	**0.65**	**0.88**	0.97	0.85	0.82
Method		***peppers*** (256×256) type **3** blur							***mandrill*** (512×512) type **3** blur						
FISTA	iterations	4	5	6	7	38	87	118	6	8	43	125	206.6	309.5	420.1
	computation time	**0.07**	**0.08**	**0.10**	0.11	0.55	1.21	1.66	**0.51**	**0.67**	3.27	9.34	15.39	23.16	31.21
SALSA	iterations	12	7	4	3	11	23	20	8	5	34	108	130	130	106
	computation time	0.29	0.17	0.12	**0.10**	**0.26**	0.50	0.44	0.98	0.68	3.60	11.06	13.77	14.68	11.95
PCD-SESOP-7	iterations	109.4	68	39	22.6	23.3	46.4	64.2	70.8	39.3	26.2	38.4	77.7	112.1	151.9
	computation time	5.05	3.15	1.79	1.04	1.08	2.15	2.95	16.01	9.19	6.11	8.94	17.42	24.89	33.92
i-LET	iterations	**3**	**3**	**2.1**	**2.2**	**8.4**	**10.6**	**10.6**	**4**	**4**	**9.6**	**19.2**	**22.3**	**23.4**	**23.7**
	computation time	0.14	0.14	**0.10**	**0.10**	0.33	**0.41**	**0.41**	0.85	0.85	**1.89**	**3.61**	**4.23**	**4.45**	**4.50**

NOTE: Iteration numbers and computation time (in sec) is averaged over ten runs. Stopping criterion: within 40 dB difference of the exact minimum of (3) (see, (16)). If the algorithm does not meet the stopping criterion within the maximum iterations allowed in some of the ten runs, we mark the results with $(*)$.

bases for the *i*-LET algorithm[6]:

$$
\begin{aligned}
\mathbf{F}_1 &= \mathbf{c}^{(n-1)},\\
\mathbf{F}_2 &= \mathbf{c}^{(n)},\\
\mathbf{F}_3 &= \overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big)\\
&= \mathbf{c}^n - \theta_{\lambda\tau/2}\Big(\mathbf{c}^{(n)} - \tau\mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\big(\mathbf{H}\mathbf{W}\mathbf{c}^{(n)} - \mathbf{y}\big)\Big),\\
\mathbf{F}_4 &= \big(\mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{W} + \mu_1\mathbf{I}\big)^{-1}\,\overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big),\\
\mathbf{F}_5 &= \big(\mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{W} + \mu_2\mathbf{I}\big)^{-1}\,\overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big).
\end{aligned}
$$

We choose τ in $\mathbf{F}_3$, $\mathbf{F}_4$ and $\mathbf{F}_5$ based on the regularization weight λ as well as the wavelet coefficients range[7], $\mathbf{c}_{\max}$, as $\tau = \mathbf{c}_{\max}/\lambda$. The values obtained by this choice are usually between 10 and 5000, much larger than 1, which is the value allowed for the convergence of the IST. We also use $\mu_1 = 1/\tau$, and $\mu_2 = 10/\tau$ to sidestep the need to select a single parameter μ empirically (see the example in Section III-C3). Notice that here we do not need to finely adjust these parameters as the LET optimization (11) automatically performs fine tuning. An IRLS inner loop with 5 iterations is used to solve the LET coefficients.

Since the reconstructed images are indistinguishable both visually and in the SNR sense, we report the iteration numbers required as well as the computational time for each algorithm in Table V. In almost all cases, the *i*-LET deconvolution algorithm outperforms FISTA, SALSA and PCD-SESOP in terms of iteration numbers. Arguably, to be meaningful in practice, we should compare the actual computational time consumed by each algorithm instead of the iterations required. In our current implementation, each *i*-LET iteration takes about 2.6 times the execution time per FISTA/SALSA iteration with the settings specified previously (i.e. LET bases and maximum IRLS inner loop iteration numbers). Hence, in most cases, the proposed *i*-LET algorithm is faster than FISTA and SALSA and in many instances, the speed improvement is quite substantial.

Notice that the *i*-LET algorithm has rather consistent performance for both high noise level and low noise level cases with various convolution kernels and test images. Arguably, this is a direct consequence of the LET optimization (11): each LET basis element is automatically (and optimally) weighted depending on the experimental configuration. We need to point out that most of the time consumption is due to the IRLS inner loop which computes the optimal LET coefficients at each iteration (see the detailed analysis in Section IV-C). With better alternatives than IRLS to solve (11), the computational time is likely to be reduced. In addition, we may have further gains with a more careful selection of LET basis.

[6]A demo software is available at http://www.ee.cuhk.edu.hk/~tblu/monsite/phps/iletdeconv.php

[7]This range is the maximum possible absolute value of the wavelet coefficients of an image, given the bounds of its pixel values; e.g. for an 8-bit gray-scale image with `haar` wavelet transform, where the decomposition filter is $\mathbf{G} = \frac{1}{\sqrt{2}}[1, -1]$ and the pixel values are in $[0, 255]$, then $\mathbf{c}_{\max} = \frac{255}{\sqrt{2}} \approx 180$.

TABLE VI
IMAGE DECONVOLUTION WITH REDUNDANT WAVELET TRANSFORM FOR *cameraman* (SYM8, THREE STAGES)

BSNR		**25**	**30**	**35**	**40**
Method		***cameraman*** type **1** blur			
FISTA	iterations	640	733	864.2	879.8
	computation time	*60.11*	*68.86*	*81.89*	*82.8*
SALSA	iterations	880	575	410	252
	computation time	*143.04*	*93.52*	*66.98*	*40.96*
PCD-SESOP-7	iterations	636.5	723.6	864	891.4
	computation time	*195.23*	*222.42*	*265.39*	*273.87*
i-LET	iterations	**154**	**113.6**	**91.9**	**67.4**
	computation time	***34.88***	***25.63***	***20.77***	***15.17***
Method		***cameraman*** type **2** blur			
FISTA	iterations	154	111	98	88
	computation time	*14.47*	*10.56*	*9.25*	*8.38*
SALSA	iterations	83	25	10	**5**
	computation time	*13.62*	*4.24*	*1.80*	***0.95***
PCD-SESOP-7	iterations	134.2	98.5	103.6	119.5
	computation time	*41.51*	*30.56*	*32.15*	*37.1*
i-LET	iterations	**23.4**	**7.5**	**5.7**	**5**
	computation time	***5.36***	***1.84***	***1.42***	*1.25*
Method		***cameraman*** type **3** blur			
FISTA	iterations	30.2	58	145	208.7
	computation time	*2.87*	*5.53*	*13.67*	*19.62*
SALSA	iterations	11	15	42	45
	computation time	*1.93*	*2.6*	*7*	*7.43*
PCD-SESOP-7	iterations	69.7	98.8	159.9	240.4
	computation time	*21.63*	*30.63*	*49.56*	*74.45*
i-LET	iterations	**7.5**	**8.4**	**14.4**	**16.4**
	computation time	***1.83***	***2.05***	***3.38***	***3.84***

NOTE: Iteration numbers and computation time (in sec) is averaged over ten runs. Stopping criterion: within 40 dB difference of the exact minimum of (3) (see, (16)).

B. Deconvolution with redundant wavelet transform

We also consider image deconvolution problems with redundant wavelet transform (RWT), which is known to provide better reconstruction quality (see Figure 7 for an example) [5], [6], [28], [56]. All the other settings remain the same with DWT deconvolution cases, including termination criterion, the way to determine proper regularization weight λ, LET bases and maximum inner IRLS iterations. However, the solution to RWT deconvolution is trivial if we use only high-pass subbands in the ℓ_1 regularization of the objective functional (3), as it is possible to deconvolve the low-pass exactly—albeit with a very unsatisfactory solution. For this reason, and in agreement with the literature [40], we choose to minimize the objective functional that includes both wavelet and low-pass subbands in the ℓ_1 regularization. Obtaining the global minimizer for the RWT deconvolution cases is obviously much slower because of the larger computational cost. We report a few representative test cases with *cameraman* in Table VI. In all but one cases, *i*-LET takes much less iterations and time to reach the stopping criterion compared to FISTA, SALSA and PCD-SESOP.

It is peculiar that all three algorithms require significantly more iterations to reach convergence in the 9×9 uniform blur cases with our current experimental configurations. One possible explanation might be that three wavelet analysis levels may not be sufficient for the 9×9 uniform blur kernel, which has 9 zeros uniformly positioned from 0 to 2π in the frequency domain: inside the low-pass subband, the convolution filter

(a) Blurred measurements (b) Reconstruction with DWT (c) Reconstruction with RWT

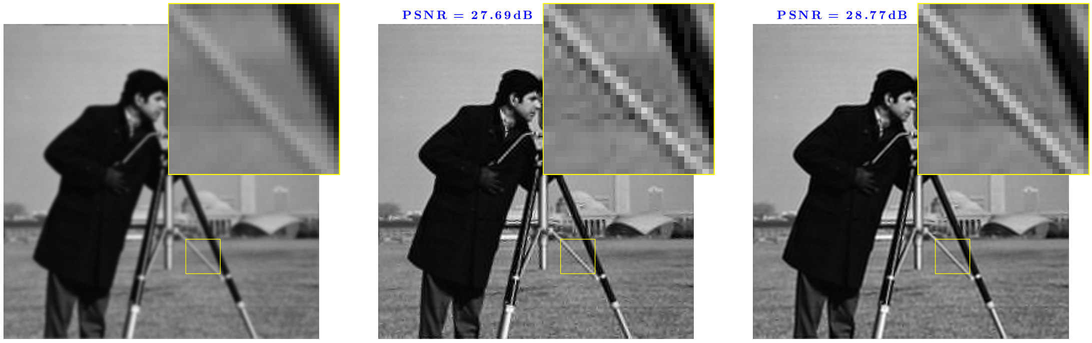


Fig. 7. Reconstructed images obtained by minimizing ℓ_1-regularized objective functional (3) with `sym8` decimated wavelet transform (DWT) and redundant wavelet transform (RWT) for *cameraman* with BSNR $= 35$ dB, convolution kernel $h_{i,j} = [1,4,6,4,1]^{\mathrm{T}}[1,4,6,4,1]/256$.

is still singular, making the thresholding of its coefficients inefficient. With increased wavelet analysis stages (e.g. 5 levels), all four algorithms take much less iterations to reach convergence for the 9×9 uniform blur cases:

BSNR		**25**	**30**	**35**	**40**
FISTA	iterations	277	319.1	378	409.9
	computation time	*42.95*	*50.19*	*58.8*	*63.89*
SALSA	iterations	169	120	84	63
	computation time	*45.98*	*32.62*	*22.91*	*17.21*
PCD-SESOP-7	iterations	384.8	400.7	456.6	505.3
	computation time	*185.12*	*192.74*	*221.89*	*244.41*
i-LET	iterations	**39.2**	**30.5**	**25.9**	**23.7**
	computation time	***14.02***	***10.89***	***9.22***	***8.48***

NOTE: Iteration numbers and computation time (in sec) is averaged over ten runs.

C. Algorithm complexity analysis

A good indication of the computational cost of the *i*-LET algorithm is its cost per iteration. At each iteration, we need to build the LET basis first and then compute the LET coefficients by optimizing the objective functional as in (11), whose solution is obtained with an IRLS inner loop. The primary cost of the algorithms is due to wavelet analysis/synthesis, point-wise thresholding functions, and to the building of the IRLS matrix $\mathbf{M}^{(n)}$ in (10). Take the LET bases for decimated wavelet deconvolution cases in Section IV-A as an example for the complexity analysis. In that case, the wavelet coefficients $\mathbf{c}$ and the measurements $\mathbf{y}$ are vectors of length N, and the LET basis is a matrix of size $N \times 5$.

The computational cost is $\mathcal{O}(N)$ to build $\mathbf{F}_1$, which depends linearly on the previous iteration results. The matrix multiplication of the convolution matrix in the generalized gradient $\overline{\nabla}_\tau J$ can be efficiently implemented in Fourier with $\mathcal{O}(N \log N)$ cost. And the wavelet transformation is of $\mathcal{O}(N)$ complexity. Since the component-wise subtraction and soft-thresholding are $\mathcal{O}(N)$ operations, the overall cost to build $\mathbf{F}_2$ is $\mathcal{O}(N \log N)$. As to $\mathbf{F}_3$, the computational cost is similar to $\mathbf{F}_2$ except for the extra preconditioning matrix, which can be easily diagonalized in Fourier with additional $\mathcal{O}(N)$ cost by applying the matrix inversion lemma:

$$\mathbf{F}_3 = \frac{1}{\mu}\big(\mathbf{I} - \mathbf{W}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}(\mu\mathbf{I} + \mathbf{H}^{\mathrm{T}}\mathbf{H})^{-1}\mathbf{H}\mathbf{W}\big)\,\overline{\nabla}_\tau\, J\big(\mathbf{c}^{(n)}\big).$$

Therefore, it takes $\mathcal{O}(N \log N)$ to build $\mathbf{F}_3$.

The IRLS matrix $\mathbf{M}^{(n)}$ involves two matrix multiplications: $(\mathbf{HWF})^{\mathrm{T}}(\mathbf{HWF})$ and $\mathbf{F}^{\mathrm{T}}\mathbf{D}^{(n)}\mathbf{F}$. Since $\mathbf{HWF}$ is an $N \times 5$ matrix, it takes $\mathcal{O}(25N)$ cost, i.e. linear-cost, to calculate $(\mathbf{HWF})^{\mathrm{T}}(\mathbf{HWF})$. While for $\mathbf{F}^{\mathrm{T}}\mathbf{D}^{(n)}\mathbf{F}$, it also has $\mathcal{O}(N)$ cost, since $\mathbf{D}^{(n)}$ is a diagonal matrix and the multiplication between $\mathbf{D}^{(n)}$ and $\mathbf{F}$ corresponds to a component-wise product that is of $\mathcal{O}(N)$ complexity. Such cost may not appear to be very expensive from the theoretical point of view. However, in practice, the matrix multiplication corresponds to loading image-size data ($\mathbf{F}$ is a 65536×5 matrix for a typical 256×256 image) from memory several times (here we use 5 IRLS inner loop iterations). That is where *i*-LET costs the most compared to other algorithms, like FISTA and SALSA. Once we have the IRLS matrix $\mathbf{M}^{(n)}$, the cost to inverse a 5×5 matrix in (9) is negligible.

D. Choice of regularization weight λ

The unconstrained problem (4) has the same solution as the constrained problem (1) for some regularization parameter $\lambda = \lambda_{\mathrm{opt}}$ in the objective functional (3). Such λ_{opt} can be obtained with a simple strategy: at each *i*-LET iteration, the regularization weight λ is updated based on the ratio between the data-fidelity and the noise energy threshold ε^2 as $\lambda^{(n+1)} = \frac{\varepsilon^2}{\|\mathbf{y}-\mathbf{HWc}^{(n)}\|_2^2}\lambda^{(n)}$ with the current iteration results $\mathbf{c}^{(n)}$ (for a similar approach in TV-based deconvolution, see [57] and an alternative in [58]). Experimentally, the algorithm converges to a solution that satisfies the constraint $\|\mathbf{y} - \mathbf{HWc}\|_2^2 \le \varepsilon^2$ at the same rate of convergence as in the case when the fixed λ_{opt} is used in the unconstrained formulation (3), i.e. we have solved the constrained problem (1) (see, Fig. 8).

V. CONCLUSION

We presented a novel approach to solve the unconstrained ℓ_1-regularized image restoration problems iteratively with linear expansion of thresholds (*i*-LET). The restoration process is represented as a linear combination of elementary restoration functions (LET basis) with unknown weights (LET coefficients). Then the linear coefficients can be computed efficiently by solving a small optimization problem in contrast to the original image-size one. The proposed *i*-LET is more a generic algorithmic framework than a specific algorithm that solves the unconstrained problem (4). As long as the basis

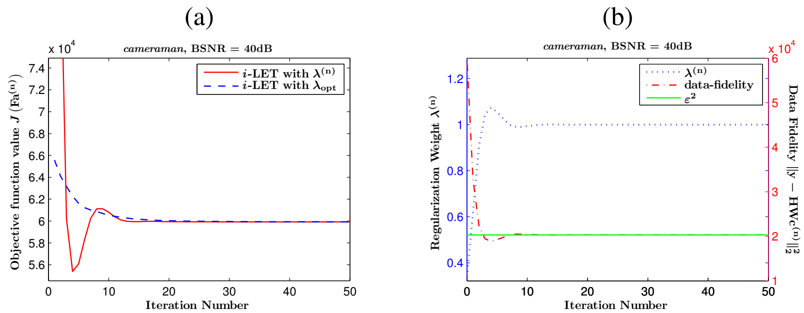


Fig. 8. (a) Evolution of the objective function value with updated λ and the fixed optimal regularization weight λ_{opt}. (b) Evolution of the regularization weight and the data-fidelity—see Section IV-D. (convolution kernel: $h_{i,j} = 1/(1 + i^2 + j^2)$ for $i, j = -7, \dots, 7$)

satisfies conditions in Section III-A, global convergence is guaranteed. Such a nice property is powerful: we can add whatever thresholding basis that has fast convergence and improve the overall speed of the new algorithm within the iterative LET framework. Variable splitting technique [35], [36], might provide such an alternative set of LET bases. Future work includes extending the proposed iterated LET framework to solve the analysis formulation problems, e.g. total-variation based image restorations.

Appendix A
Proofs and Derivations

A. Proof of Lemma 2

Take a small variation $\delta\mathbf{c}$ from $\mathbf{c}$ then,

$$\begin{aligned} &J(\mathbf{c} + \delta\mathbf{c}) - J(\mathbf{c}) \\ &= J_0(\mathbf{c} + \delta\mathbf{c}) - J_0(\mathbf{c}) + \lambda\|\mathbf{c} + \delta\mathbf{c}\|_1 - \lambda\|\mathbf{c}\|_1 \\ &= \|\mathbf{A}\delta\mathbf{c}\|_2^2 + \delta\mathbf{c}^{\mathrm{T}}\nabla J_0 + \lambda\|\mathbf{c} + \delta\mathbf{c}\|_1 - \lambda\|\mathbf{c}\|_1 \end{aligned} \tag{17}$$

Let $J_1(\delta\mathbf{c}) = \dfrac{1}{\tau}\|\delta\mathbf{c}\|_2^2 + \delta\mathbf{c}^{\mathrm{T}}\nabla J_0 + \lambda\|\mathbf{c} + \delta\mathbf{c}\|_1 - \lambda\|\mathbf{c}\|_1$, we have

$$J(\mathbf{c} + \delta\mathbf{c}) - J(\mathbf{c}) = J_1(\delta\mathbf{c}) + \|\mathbf{A}\delta\mathbf{c}\|_2^2 - \frac{1}{\tau}\|\delta\mathbf{c}\|_2^2.$$

Notice that we can rewrite $J_1(\delta\mathbf{c})$ as

$$\begin{aligned} J_1(\delta\mathbf{c}) =& \frac{1}{\tau}\left\|\delta\mathbf{c} + \frac{\tau}{2}\nabla J_0\right\|_2^2 + \lambda\|\mathbf{c} + \delta\mathbf{c}\|_1 \\ &+ \text{terms independent of } \delta\mathbf{c} \end{aligned}$$

Hence,

$$\arg\min_{\delta\mathbf{c}} J_1(\delta\mathbf{c}) = \arg\min_{\delta\mathbf{c}} \left\{\frac{1}{\tau}\left\|\delta\mathbf{c} + \frac{\tau}{2}\nabla J_0\right\|_2^2 + \lambda\|\mathbf{c} + \delta\mathbf{c}\|_1\right\} \tag{18}$$

The solution to (18) is given by the standard soft-thresholding [59]: $\delta\mathbf{c} = -\mathbf{c} + \theta_{\lambda\tau/2}\left(\mathbf{c} - \frac{\tau}{2}\nabla J_0\right) = -\frac{\tau}{2}\overline{\nabla}_\tau J$. Since $J_1(\delta\mathbf{c})$ is a strictly convex function, we have

$$\begin{aligned} J_1(-\varepsilon\overline{\nabla}_\tau J) &= J_1\Big(\big(1 - 2\varepsilon/\tau\big)\mathbf{0} + 2\varepsilon/\tau\big(-\tfrac{\tau}{2}\overline{\nabla}_\tau J\big)\Big) \\ &< \big(1 - 2\varepsilon/\tau\big)J_1(\mathbf{0}) + 2\varepsilon/\tau J_1(-\tfrac{\tau}{2}\overline{\nabla}_\tau J) \\ &= 2\varepsilon/\tau J_1(-\tfrac{\tau}{2}\overline{\nabla}_\tau J) \end{aligned} \tag{19}$$

for $-\frac{\tau}{2}\overline{\nabla}_\tau J \neq \mathbf{0}, \varepsilon \in (0, \tau/2)$. On the other hand, from (18) we have $J_1(-\frac{\tau}{2}\overline{\nabla}_\tau J) \leq J_1(\mathbf{0}) = 0$. Combined with (19), we have $J_1(-\varepsilon\overline{\nabla}_\tau J) < 0$. Therefore,

$$\begin{aligned} J(\mathbf{c} - \varepsilon\overline{\nabla}_\tau J) - J(\mathbf{c}) &= J_1(-\varepsilon\overline{\nabla}_\tau J) + \|\varepsilon\mathbf{A}\overline{\nabla}_\tau J\|_2^2 - \frac{1}{\tau}\|\varepsilon\overline{\nabla}_\tau J\|_2^2 \\ &< 2\varepsilon/\tau J_1\big(-\tfrac{\tau}{2}\overline{\nabla}_\tau J\big) + O(\varepsilon^2) < 0 \end{aligned}$$

for sufficiently small $\varepsilon > 0$. We can further reduce the objective functional value along the negative direction of the generalized gradient $-\overline{\nabla}_\tau J$ unless $\overline{\nabla}_\tau J = \mathbf{0}$, i.e. J is at the minimum.

B. Proof of Theorem 2

We first have to prove the decrease of the criterion $J(\mathbf{c})$ down to its *global* minimum. In order to do this, we consider the sequence of iterates $\mathbf{c}^{(n)}$, $n \in \mathbb{N}$, provided by Algorithm 1. We can make the following observations

1) $J(\mathbf{c}^{(n)})$ is a decreasing positive sequence, hence it *converges* to some value J_∞ as $n \to \infty$;
2) for any $\varepsilon \in \mathbb{R}$ and any index $n' \geq n + 1$, $J(\mathbf{c}^{(n')}) \leq J(\mathbf{c}^{(n+1)}) \leq J(\mathbf{c}^{(n)} - \varepsilon\overline{\nabla}_\tau(\mathbf{c}^{(n)}))$ (because $\mathbf{c}^{(n)}$ and $\overline{\nabla}_\tau(\mathbf{c}^{(n)}$ are two LET basis elements, from which $\mathbf{c}^{(n+1)}$ is optimally built);
3) $\mathbf{c}^{(n)}$ belongs to the *closed bounded* ball $\|\mathbf{c}^{(n)}\|_1 \leq \lambda^{-1}J(\mathbf{c}^{(0)})$;
4) closed bounded balls of $\mathbb{R}^D$ are *compact*, which implies that a *convergent subsequence* can be extracted from $\mathbf{c}^{(n)}$ (Bolzano-Weierstrass Theorem).

Let us then denote by n_k, $k \in \mathbb{N}$, the indices for which the subsequence $\mathbf{c}^{(n_k)}$ is convergent as $k \to \infty$, and let us call $\mathbf{c}_\infty = \lim_{k\to\infty}\mathbf{c}^{(n_k)}$. According to point 2) above, the subsequence satisfies the inequality

$$\text{for any } \varepsilon \in \mathbb{R}, \quad J(\mathbf{c}^{(n_{k+1})}) \leq J(\mathbf{c}^{(n_k)} - \varepsilon\overline{\nabla}_\tau(\mathbf{c}^{(n_k)})).$$

Now, we can pass to the limit as $k \to \infty$: since J and $\overline{\nabla}_\tau$ are *continuous* we have that

- $J(\mathbf{c}_\infty) = J(\lim_{k\to\infty}\mathbf{c}^{(n_k)}) = \lim_{k\to\infty} J(\mathbf{c}^{(n_k)}) = J_\infty$;
- $J_\infty = \lim_{k\to\infty} J(\mathbf{c}^{(n_{k+1})}) \leq \lim_{k\to\infty} J(\mathbf{c}^{(n_k)} - \varepsilon\overline{\nabla}_\tau(\mathbf{c}^{(n_k)})) = J(\mathbf{c}_\infty - \varepsilon\overline{\nabla}_\tau(\mathbf{c}_\infty))$.

The last inequality, satisfied for any real value of ε is particularly interesting because we know, from Lemma 2 that the generalized gradient provides a *strictly decreasing* direction for J. This means that, if $\overline{\nabla}_\tau J(\mathbf{c}_\infty) \neq \mathbf{0}$, we can choose ε in such a way that

$$J_\infty \leq J(\mathbf{c}_\infty - \varepsilon\overline{\nabla}_\tau(\mathbf{c}_\infty)) < J(\mathbf{c}_\infty) = J_\infty,$$

which is impossible. Hence, we conclude that $\overline{\nabla}_\tau J(\mathbf{c}_\infty) = \mathbf{0}$, and thus that $J_\infty = \min_{\mathbf{c}\in\mathbb{R}^D} J(\mathbf{c})$ (by Lemma 1). To summarize, we have proven that the iterative LET scheme provides a global minimization of the criterion (3); i.e.,

$$\lim_{n\to\infty} J(\mathbf{c}^{(n)}) = \min_{\mathbf{c}\in\mathbb{R}^D} J(\mathbf{c}).$$

The second thing we have to prove is that, if we know that the minimizer of $J(\mathbf{c})$ is unique (e.g., by checking the sufficient conditions of Theorem 1), then the sequence of *i*-LET iterates converges to that minimizer. We consider again a convergent subsequence of $\mathbf{c}^{(n)}$: according to the above proof, its limit $\mathbf{c}_\infty$ should satisfy $\overline{\nabla}_\tau J(\mathbf{c}_\infty) = \mathbf{0}$; i.e., $\mathbf{c}_\infty$ is a minimizer of $J(\mathbf{c})$. But now, we know that such a minimizer is unique, hence we can say that all the convergent subsequences have the same limit, or, said differently: all the *limit points* of the *i*-LET sequence coincide. Since this sequence belongs to a compact space, we can conclude that the sequence converges to this unique value.

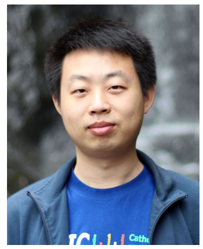
ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their constructive comments and suggestions. This work was supported by an RGC grant #CUHK410110 of the Hong Kong University Grant Council.

REFERENCES

[1] H. Pan and T. Blu, "Sparse image restoration using iterated linear expansion of thresholds," in *2011 IEEE International Conference on Image Processing*. IEEE, 2011, pp. 1905–1908.
[2] H. Engl, M. Hanke, and A. Neubauer, *Regularization of inverse problems*. Springer, 1996, vol. 375.
[3] J. Cai, R. Chan, and M. Nikolova, "Two-phase approach for deblurring images corrupted by impulse plus Gaussian noise," *Inverse Problems and Imaging*, vol. 2, no. 2, pp. 187–204, 2008.
[4] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004.
[5] M. Elad, "Why simple shrinkage is still relevant for redundant representations?" *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5559–5569, 2006.
[6] M. Elad, B. Matalon, and M. Zibulevsky, "Image denoising with shrinkage and redundant representations," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2006, pp. 1924–1931.
[7] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33–61, 1999.
[8] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
[9] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, 2010.
[10] J. Starck, F. Murtagh, and J. Fadili, *Sparse image and signal processing: wavelets, curvelets, morphological diversity*. Cambridge University Press, 2010.
[11] M. Zibulevsky and M. Elad, "L1-L2 optimization in signal and image processing," *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 76–88, 2010.
[12] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.
[13] T. Chan and C. Wong, "Total variation blind deconvolution," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 370–375, 1998.
[14] M. Elad, P. Milanfar, and R. Rubinstein, "Analysis versus synthesis in signal priors," *Inverse Problems*, vol. 23, p. 947, 2007.
[15] I. Selesnick and M. Figueiredo, "Signal restoration with overcomplete wavelet transforms: comparison of analysis and synthesis priors," in *Proceedings of SPIE*, 2009, pp. 74 460D–74 460D.
[16] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale ℓ_1-regularized least squares," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 606 –617, 2007.
[17] Y. Nesterov, "Smooth minimization of non-smooth functions," *Mathematical Programming*, vol. 103, no. 1, pp. 127–152, 2005.
[18] ——, "Gradient methods for minimizing composite objective function," *CORE Discussion Papers*, no. 2007/76, 2007. [Online]. Available: http://www.optimization-online.org
[19] E. Hale, W. Yin, and Y. Zhang, "Fixed-point continuation for ℓ_1-minimization: Methodology and convergence," *SIAM Journal on Optimization*, vol. 19, no. 3, pp. 1107–1130, 2008.
[20] M. Figueiredo, R. Nowak, and S. Wright, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 586–597, 2007.
[21] P. Combettes and V. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.
[22] N. Pustelnik, C. Chaux, and J. Pesquet, "Parallel proximal algorithm for image restoration using hybrid regularization," *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2450–2462, 2011.
[23] J. Fadili and J. Starck, "Sparse representation-based image deconvolution by iterative thresholding," *Astronomical Data Analysis*, vol. 6, p. 18, 2006.
[24] F. Dupé, J. Fadili, and J. Starck, "A proximal iteration for deconvolving Poisson noisy images using sparse representations," *IEEE Transactions on Image Processing*, vol. 18, no. 2, pp. 310–321, 2009.
[25] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of Mathematical Imaging and Vision*, vol. 40, no. 1, pp. 120–145, 2011.
[26] C. Vonesch and M. Unser, "A fast iterative thresholding algorithm for wavelet-regularized deconvolution," in *Proceedings of SPIE*, vol. 6701, 2007, p. 67010D.
[27] M. Figueiredo and R. Nowak, "An EM algorithm for wavelet-based image restoration," *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 906–916, 2003.
[28] M. Figueiredo, J. Bioucas-Dias, and R. Nowak, "Majorization-minimization algorithms for wavelet-based image restoration," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2980–2991, 2007.
[29] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.
[30] K. Bredies and D. Lorenz, "Linear convergence of iterative soft-thresholding," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5, pp. 813–837, 2008.
[31] J. Bioucas-Dias and M. Figueiredo, "A new TwIST: two-step iterative shrinkage/thresholding algorithms for image restoration," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 2992–3004, 2007.
[32] G. Narkiss and M. Zibulevsky, *Sequential subspace optimization method for large-scale unconstrained problems*. Technion-IIT, Department of Electrical Engineering, 2005.
[33] M. Elad, B. Matalon, and M. Zibulevsky, "Coordinate and subspace optimization methods for linear least squares with non-quadratic regularization," *Applied and Computational Harmonic Analysis*, vol. 23, no. 3, pp. 346–367, 2007.
[34] E. Chouzenoux, J. Idier, and S. Moussaoui, "A majorize–minimize strategy for subspace optimization applied to image restoration," *IEEE Transactions on Image Processing*, vol. 20, no. 6, pp. 1517–1528, 2011.
[35] R. Courant, "Variational methods for the solution of problems of equilibrium and vibrations," *Bulletin of American Mathematical Society*, vol. 49, pp. 1–23, 1943.
[36] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 248–272, 2008.
[37] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1-problems in compressive sensing," *SIAM Journal on Scientific Computing*, vol. 33, no. 1, pp. 250–278, 2011.
[38] T. Goldstein and S. Osher, "The split Bregman method for L1-regularized problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, 2009.
[39] S. Wright, R. Nowak, and M. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.

[40] M. Afonso, J. Bioucas-Dias, and M. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2345–2356, 2010.
[41] ——, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 681–695, 2011.
[42] F. Luisier, T. Blu, and M. Unser, "A new SURE approach to image denoising: Interscale orthonormal wavelet thresholding," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 593–606, 2007.
[43] T. Blu and F. Luisier, "The SURE-LET approach to image denoising," *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2778–2786, 2007.
[44] D. Hunter and K. Lange, "A tutorial on MM algorithms," *The American Statistician*, vol. 58, no. 1, pp. 30–37, 2004.
[45] D. Wipf and S. Nagarajan, "Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 317–329, 2010.
[46] M. Osborne, *Finite algorithms in optimization and data analysis*. John Wiley & Sons, NY, 1985.
[47] I. Daubechies, R. DeVore, M. Fornasier, and C. Güntürk, "Iteratively reweighted least squares minimization for sparse recovery," *Communications on Pure and Applied Mathematics*, vol. 63, no. 1, pp. 1–38, 2010.
[48] I. Gorodnitsky and B. Rao, "Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm," *IEEE Transactions on Signal Processing*, vol. 45, no. 3, pp. 600–616, 1997.
[49] J. Bioucas-Dias, "Bayesian wavelet-based image deconvolution: a GEM algorithm exploiting a class of heavy-tailed priors," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 937–951, 2006.
[50] T. Chandrupatla and A. Belegundu, *Introduction to finite elements in engineering*. Prentice-Hall Englewood Cliffs, NJ, 1991.
[51] D. Donoho and I. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.
[52] Z. Luo and P. Tseng, "On the convergence of the coordinate descent method for convex differentiable minimization," *Journal of Optimization Theory and Applications*, vol. 72, no. 1, pp. 7–35, 1992.
[53] M. Osborne, B. Presnell, and B. Turlach, "On the LASSO and its dual," *Journal of Computational and Graphical Statistics*, vol. 9, no. 2, pp. 319–337, 2000.
[54] J. Fuchs, "Recovery of exact sparse representations in the presence of bounded noise," *IEEE Transactions on Information Theory*, vol. 51, no. 10, pp. 3601–3608, 2005.
[55] E. Candès and Y. Plan, "Near-ideal model selection by ℓ_1 minimization," *The Annals of Statistics*, vol. 37, no. 5A, pp. 2145–2177, 2009.
[56] J. L. Starck, M. Elad, and D. Donoho, "Redundant multiscale transforms and their application for morphological component separation," *Advances in Imaging and Electron Physics*, vol. 132, pp. 287–348, 2004.
[57] J. Oliveira, J. Bioucas-Dias, and M. Figueiredo, "Adaptive total variation image deblurring: a majorization-minimization approach," *Signal Processing*, vol. 89, no. 9, pp. 1683–1693, 2009.
[58] Y. Wen and R. Chan, "Parameter selection for total variation based image restoration using discrepancy principle," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1770 –1781, 2012.
[59] A. Chambolle, R. De Vore, N. Lee, and B. Lucier, "Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 319–335, 1998.



Hanjie Pan (S'11) was born in Jiangsu, China, in 1988. In 2010, he received the B.Eng. degree with first honor in Electronic Engineering from The Chinese University of Hong Kong (CUHK), Shatin, Hong Kong. He received the M.Phil degree from the same institution in 2013. In the summer 2010, he was a visiting student at Imperial College, London, United Kingdom. His research interests include image restorations with sparsity constraints and sampling signals with finite rate of innovation.

Thierry Blu (F'12) was born in Orléans, France, in 1964. He received the "Diplôme d'ingénieur" from École Polytechnique, France, in 1986 and from Télécom Paris (ENST), France, in 1988. In 1996, he obtained a Ph.D in electrical engineering from ENST for a study on iterated rational filterbanks, applied to wideband audio coding.

Between 1998 and 2007, he was with the Biomedical Imaging Group at the Swiss Federal Institute of Technology (EPFL) in Lausanne, Switzerland. He is now a Professor in the Department of Electronic Engineering, The Chinese University of Hong Kong.

Dr. Blu was the recipient of two best paper awards from the IEEE Signal Processing Society (2003 and 2006). He is also coauthor of a paper that received a Young Author best paper award (2009) from the same society. He has been an Associate Editor for the IEEE Transactions on Image Processing (2002–2006), the IEEE Transactions on Signal Processing (2006–2010), Elsevier Signal Processing (2008–2011). He is currently on the board of EURASIP J. on Image and Video Processing (since 2010) and is a member of the IEEE Signal Processing Theory and Methods Technical Committee (since 2008).

Research interests: (multi)wavelets, multiresolution analysis, multirate filterbanks, interpolation, approximation and sampling theory, sparse sampling, image denoising, psychoacoustics, biomedical imaging, optics, wave propagation...