IEEE Trans. on Image Processing, 2014, 23(12), pp. 5274–5283

Fast Ordering Algorithm for Exact Histogram Specification

Mila Nikolova (Senior Member IEEE), and Gabriele Steidl

***Abstract*—This paper provides a fast algorithm to order in a meaningful, strict way the integer gray values in digital (quantized) images. It can be used in any exact histogram specification based application. Our algorithm relies on the ordering procedure based on the specialized variational approach proposed in [1]. This variational method was shown to be superior to all other state-of-the art ordering algorithms in terms of faithful total strict ordering but not in speed. Indeed, the relevant functionals are in general difficult to minimize because their gradient is nearly flat over vast regions.**

In this paper we propose a simple and fast fixed point algorithm to minimize these functionals. The fast convergence of our algorithm results from known analytical properties of the model. Our algorithm is equivalent to an iterative nonlinear filtering. Further we show that a particular form of the variational model gives rise to much faster convergence than other alternative forms. We demonstrate that only a few iterations of this filter yield almost the same pixel ordering as the minimizer. Thus we apply only few iteration steps to obtain images whose pixels can be ordered in a strict and faithful way.

Numerical experiments confirm that our algorithm outperforms by far its main competitors.

***Index Terms*—Exact histogram specification, strict ordering, variational methods, fully smoothed L_1-TV models, nonlinear filtering, fast convex minimization**

I. Introduction

Histogram processing is a technique with numerous applications, e.g., in image normalization and enhancement, object recognition, and invisible watermarking, [2]–[5]. The goal of exact histogram specification (HS) is to transform an input image into an output image having a prescribed histogram. For a uniform target histogram we speak about histogram equalization (HE).

Consider digital (i.e. quantized) $M \times N$ images f with L gray values $\mathcal{Q} := \{q_1, \cdots, q_L\}$.. For 8-bit images we have $L = 256$ and $\mathcal{Q} = \{0, \cdots, 255\}$. We reorder the image columnwise into a vector of size $n := MN$ and address the pixels by the index set $\mathbb{I}_n := \{1, \cdots, n\}$. The histogram of f, denoted by h_f, is given by $h_f[q_k] = \sharp\{i \in \mathbb{I}_n \mid f[i] = q_k\}$, $k = 1, \ldots, L$, where $\sharp$ stands for cardinality.

In theory, histogram specification uses the relation between the cumulative density function of an arbitrary distributed continuous random variable and a uniformly distributed one, see [3]. However, for digital images we are confronted with a large number of n discrete variables taking only L possible values (i.e., $n \gg L$). Then the target histogram can almost never be satisfied exactly. Histogram specification is an ill-posed problem for digital images. The Matlab function `histeq` is expected to produce HE but it usually fails. The importance of a meaningful strict ordering of pixels prior to histogram specification is illustrated in Fig. 1; see the comments in the caption.

In this paper we focus on *exact* histogram specification to a target histogram $\widehat{h} = (\widehat{h}_1, \ldots, \widehat{h}_L)$ for the gray values $\mathcal{P} = \{0, \ldots, L-1\}$. If the pixels values of our image are pairwise different so that they can be strictly ascending ordered, exact histogram specification can be easily done by dividing the corresponding ordered list of indices into L groups and assigning gray value 0 to the first $\widehat{h}_1$ pixels, gray value 1 to the second $\widehat{h}_2$ pixels and so on until gray value $L-1$ is assigned to the last $\widehat{h}_L$ pixels. This simple procedure yields good results if one has a *meaningful strict ordering* of all pixels in the input image. Fig. 1 demonstrates the importance of ordering for histogram equalization.

Research on this problem has been conducted for four decades already [7]. State-of-the-art methods are

- the local mean ordering (LM) of Coltuc, Bolon and Chassery [8],
- the wavelet-based ordering (WA) of Wan and Shi [9],
- the variational approach (VA) of Nikolova, Wen and Chan [1] based on the minimization of a fully smoothed ℓ_1-TV functional.

The first two methods extract for any pixel $f[i]$ in the input image K auxiliary informations, say $\kappa_k[i]$, $k \in \mathbb{I}_K$, where $\kappa_1 := f$. Then an ascending order "$\prec$" for all pixels could ideally be obtained using the rule

$$\begin{aligned} & i \prec j \quad \text{if} \quad \kappa_s[i] < \kappa_s[j] \text{ for some } s \in \mathbb{I}_K \\ & \text{and} \\ & \kappa_k[i] = \kappa_k[j] \quad \text{for all} \quad 1 \le k < s. \end{aligned} \tag{1}$$

The third method uses an iterative procedure to find the minimizer of a specially designed functional related to f which components can be ordered in a strict way. The numerical results in [1] have shown that VA clearly outperforms LM and WA in terms of quality of the ordering and memory requirements. However, the minimizer was computed by the Polak-Ribiére algorithm and the whole ordering algorithm was slower than LM and WA.

Contributions. We propose a simple fixed point algorithm that attains the minimizer of fully smoothed ℓ_1-TV functionals with remarkable speed and precision. In this algorithm all needed derivative and inverse functions can be given in

M. Nikolova is with the CMLA - CNRS, ENS Cachan, 61 av. President Wilson, 94235 Cachan Cedex, France (email: nikolova@cmla.ens-cachan.fr).

Her work was supported by the "FMJH Program Gaspard Monge in optimization and operation research", and by the support to this program from EDF.

G. Steidl is with the Dept. of Mathematics, University of Kaiserslautern, Paul Ehrlich Str. 31, 67663 Kaiserslautern, Germany (email: steidl@mathematik.uni-kl.de).

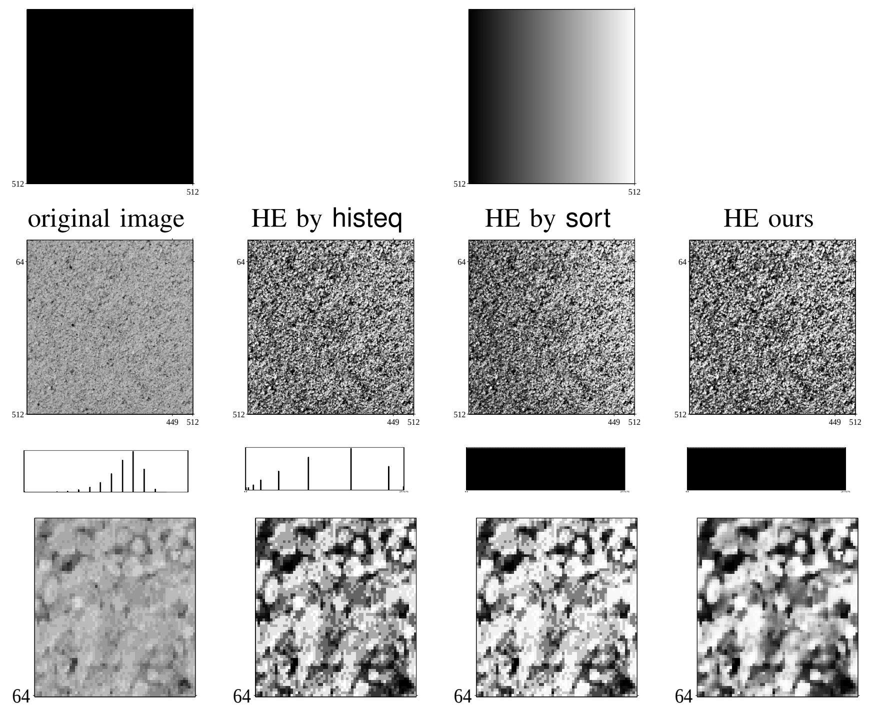

Fig. 1. A meaningful strict ordering is indeed crucial for histogram equalization (HE). First row: The pixels of a completely black image (left) are strictly ordered using the Matlab routine sort which sorts equal pixels columnwise ascending. The resultant HE image (right) is perfectly equalized and becomes lighter from left to right. Second row: The original image 'sand' (http://sipi.usc.edu/database/) of size 512×512 and different equalizations: Matlab histeq, Matlab sort and our sorting algorithm preceding the HE step (last two images). The third image still has the lighting effects from left to right. Third row: The corresponding histograms. Fourth row: Zooms of the images in the second row (rows from 1 to 64 and columns from 449 to 512). It can be seen that the texture generated by our HE algorithm is more regular and looks more natural than the other ones.

an explicit form, so our minimization scheme amounts to a particular nonlinear filter. Convergence and parameter selection are explained based on new theoretical results and on few relevant facts from [1] and [11]. The classical smooth approximation of the ℓ_1 norm is $\theta_1(t) = \sqrt{t^2+\alpha}$ for $\alpha \gtrapprox 0$; it was used in [1] together with the Polak-Ribiére algorithm, and in the conference paper [10]. Noticing that the derivatives and the inverses for θ_1 are slow to compute, we looked for other approximations where these functions are easy to compute: see θ_2 in Tab. I. We demonstrate that in our context, both approximations yield the same minimizer, up to negligible errors. Using θ_2 instead of θ_1 in the fixed point algorithm reduces the running time by 20 %. We observed that for nearly all images, a meaningful strict ordering of all pixels is obtained only after a very small number of iterations. Consequently, we use only few (e.g. 3 to 5) iterations with our nonlinear filter to get the information needed for a meaningful strict ordering. All these facts decrease the running time from paper [1] by a factor between 50 and 30! In contrast to the LM and the WA methods that extract the ordering based on K images, our algorithm requires just a single ordering of one image. Numerical tests confirm that our new algorithm, involving the function θ_2 and the needed small number of iterations, outperforms by far all other relevant ordering methods in terms of quality and speed.

As already pointed out, one can design fast HS methods based on our ordering algorithm. Therefore, the present paper provides the background for any exact HS based application, e.g., image normalization and image enhancement, among others. We have used our algorithm successfully for hue and range preserving HS based color image enhancement in [2].

Outline. In Section II we review the specialized variational approach and some of its properties proved in [1], [11] which are relevant to this work. In Section III, we propose a simple fixed point algorithm to find a minimizer of our functional. The reasons for its efficiency and effectiveness are explained. Section IV provides numerical examples. We compare speed and accuracy in the sense of a faithful total strict ordering of our algorithm with state-of-the art algorithms and provide a histogram equalization inversion comparison. Experiments clearly demonstrate that only few iterations of our algorithm are necessary to obtain promising ordering results. Conclusions are given in Section V.

II. The fully smoothed ℓ_1−TV model

Let D_N denote the forward difference matrix

$$D_N := \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & & \ddots & \\ & & & -1 & 1 \end{pmatrix} \in \mathbb{R}^{N-1,N}.$$

We will apply forward differences to the rows and columns of images, i.e., with respect to the horizontal and vertical directions. Since we consider $N \times M$ images columnwise reordered into vectors of length $n = MN$, the forward difference operator applied to these images reads as

$$G := \begin{pmatrix} I_M \otimes D_N \\ D_M \otimes I_N \end{pmatrix} \in \mathbb{R}^{r,n},$$

where I_N is the $N \times N$ identity matrix, $\otimes$ denotes the Kronecker product and $r = 2MN - M - N$. We consider functionals of the form

$$J(u, f) := \Psi(u, f) + \beta\Phi(u), \quad \beta > 0 \tag{2}$$

with

$$\begin{aligned} \Psi(u, f) &:= \sum_{i \in \mathbb{I}_n} \psi(u[i] - f[i]), \\ \Phi(u) &:= \sum_{j \in \mathbb{I}_r} \varphi\left((Gu)[j]\right). \end{aligned} \tag{3}$$

Here $(Gu)[j]$ denotes the jth component of the vector $Gu \in \mathbb{R}^r$. One could additionally use diagonal differences to improve the rotation invariance of $\Phi(u)$. However, our experiments have shown that the simple forward differences in horizontal and vertical directions are enough to enable the minimizer of J to give rise to a prompt sorting.

Following [1], the essence for achieving a strict ordering is that the functions $\psi(\cdot) := \psi(\cdot, \alpha_1)$ and $\varphi(\cdot) := \varphi(\cdot, \alpha_2)$ belong to a *family of functions* $\theta(\cdot, \alpha) : \mathbb{R} \to \mathbb{R}$, $\alpha > 0$, satisfying the requirements in assumptions H1 and H2 described next. The rationale for these choices was extensively discussed in [1]. For simplicity, the parameters α_1 and α_2 are omitted when they are not explicitly involved in our derivations.

Assumptions. *In the following, we systematically denote*

$$\theta'(t, \alpha) := \frac{d}{dt}\theta(t, \alpha) \quad \text{and} \quad \theta''(t, \alpha) := \frac{d^2}{dt^2}\theta(t, \alpha).$$

H1 *For any fixed $\alpha > 0$ the function $t \mapsto \theta(t, \alpha)$ is in $\mathcal{C}^2(\mathbb{R})$ and even, i.e., $\theta(-t, \alpha) = \theta(t, \alpha)$ for all $t \in \mathbb{R}$. Its derivative $\theta'(t, \alpha)$ is strictly increasing with $\lim_{t\to\infty} \theta'(t, \alpha) = 1$, where the upper bound is set to 1 just for definiteness. The second derivative $\theta''(t, \alpha)$ is decreasing on $[0, +\infty)$.*

H2 *For fixed $t > 0$, the function $\alpha \mapsto \theta(t, \alpha)$ is strictly decreasing on $(0, +\infty)$ with*

$$\lim_{\alpha\to 0} \theta'(t, \alpha) = 1 \quad \textit{and} \quad \lim_{\alpha\to\infty} \theta'(t, \alpha) = 0\ .$$

Under these assumptions ψ and φ are smooth convex approximations of the absolute value function. Hence the functional $J(\cdot, f)$ in (2)-(3) amounts to a fully smoothed ℓ_1-TV model.

¿From H1 it follows that $\theta'(t, \alpha)$ is odd and has an inverse function

$$\xi(t, \alpha) := (\theta')^{-1}(t, \alpha). \tag{4}$$

Clearly, $t \mapsto \xi(t, \alpha)$ is also odd and strictly increasing on $(-1, 1)$. Moreover, since $\theta''(t, \alpha)$ is positive and decreasing on $[0, +\infty)$, the function ξ is differentiable and

$$\xi'(t, \alpha) = \frac{1}{\theta''(\xi(t))} > 0. \tag{5}$$

So $t \mapsto \xi'(t, \alpha)$ is also increasing on $(0, 1)$. There are many possible choices of functions θ meeting H1 and H2, see [1]. In our numerical tests we use the functions $\theta = \psi = \varphi$ with $\alpha_1 = \alpha_2 = \alpha$ given in the following table:

	θ	θ'	$\xi = (\theta')^{-1}$	ξ'
θ_1	$\sqrt{t^2 + \alpha}$	$\frac{t}{\sqrt{t^2+\alpha}}$	$t\sqrt{\frac{\alpha}{1-t^2}}$	$\frac{\sqrt{\alpha}}{(\sqrt{1-t^2})^3}$
θ_2	$\lvert t\rvert - \alpha \log\left(1 + \frac{\lvert t\rvert}{\alpha}\right)$	$\frac{t}{\alpha+\lvert t\rvert}$	$\frac{\alpha t}{1-\lvert t\rvert}$	$\frac{\alpha}{(1-\lvert t\rvert)^2}$

TABLE I
CHOICES FOR $\theta(\cdot, \alpha)$ TOGETHER WITH THE USED DERIVATIVES AND INVERSE FUNCTIONS.

Since $J(\cdot, f)$ is a strictly convex, coercive functional it has a unique minimizer $\widehat{u} \in \mathbb{R}^n$. The next theorems summarize several properties of this minimizer which are important for our faithful and fast sorting algorithm. The first theorem proven in [1, Theorem 1] ensures that the entries of the minimizer differ in general pairwise from each other so that $\widehat{u}$ provides an auxiliary information for ordering the pixels of f.

Theorem 1. (Strict ordering information)
Let ψ and φ fulfill H1 and H2. Then there exists a dense open subset $\mathbb{K}^n$ of $\mathbb{R}^n$ *such that for any $f \in \mathbb{K}^n$ the minimizer $\widehat{u}$ of $J(\cdot, f)$ satisfies*

$$\begin{aligned} &\widehat{u}[i] \neq \widehat{u}[j], \qquad &&\forall\, i, j \in \mathbb{I}_n, \quad i \neq j, \\ &\widehat{u}[i] \neq f[i], &&\forall\, i \in \mathbb{I}_n. \end{aligned} \tag{6}$$

The fact that $\mathbb{K}^n$ is dense and open in $\mathbb{R}^n$ means that the property in (6) is generically true. This result is much stronger than saying that (6) holds true almost everywhere on $\mathbb{R}^n$. [1] So the minimizers $\widehat{u}$ that fail (6) are quite rare.

The second theorem provides an estimate of $\|f - \widehat{u}\|_\infty$ which has been proven by the authors in [11, Theorems 1 and 2].

Theorem 2. (Distance of $\widehat{u}$ from f)
Let ψ and φ fulfill H1 and H2 and let $\beta < \frac{1}{4}$. Then, for any $f \in \mathbb{R}^n$, the minimizer $\widehat{u}$ of $J(\cdot, f)$ satisfies

$$\|\widehat{u} - f\|_\infty \leq (\psi')^{-1}\big(4\beta, \alpha_1\big) = \xi\big(4\beta, \alpha_1\big), \tag{7}$$

where $\xi := (\psi')^{-1}$. Further it holds

$$\|\widehat{u} - f\|_\infty \nearrow \xi\big(4\beta, \alpha_1\big) \quad \text{as} \quad \alpha_2 \searrow 0 \tag{8}$$

if $\nu_f := \max_{i\in\mathcal{I}} \big\{ \min\big(\big|f[i] - f[i-1]\big|, \big|f[i] - f[i-M]\big|\big) > 2\xi\big(4\beta, \alpha_1\big)\big\}$, where $\mathcal{I} := \big\{i \in \operatorname{int} \mathbb{I}_n : (f[i] - f[i-1])(|f[i] - f[i-M]|) \neq 0\big\} \neq \emptyset$. Here $\operatorname{int} \mathbb{I}_n$ denotes the set of indices of non boundary pixels.

The upper bound (7) clearly indicates how to select the parameters in order to guarantee that $|f[i] - \widehat{u}[i]| < 0.5$ for any $i \in \mathbb{I}_n$. Consequently, if for $f[i] \in \{0, \ldots, 255\}$, $i \in \mathbb{I}_n$, the relation $f[i] < f[j]$ holds true, then also $\widehat{u}[i] < \widehat{u}[j]$ such that the initial ordering of pairwise different pixels is preserved. More precisely, we obtain for $\beta = 0.1$ and $\alpha_1 = \alpha_2 = 0.05$ that $\|\widehat{u} - f\|_\infty \leq 0.0976$ if $\psi = \varphi = \theta_1$ and $\|\widehat{u} - f\|_\infty \leq 0.0333$ if $\psi = \varphi = \theta_2$.

[1] An almost everywhere true property requires *only* that $\mathbb{K}^n$ is dense in $\mathbb{R}^n$. But $\mathbb{K}^n$ may not contain open subsets. There are many examples. For instance, $\mathbb{K} := [0, 1] \setminus \{x \in [0, 1] \ : \ x \text{ is rational}\}$ is dense in $[0, 1]$ and $\mathbb{K}$ does not contain open subsets.

Concerning the lower bound (8) we emphasize that the assumption on ν_f is realistic for natural images with 8 bit gray values; see [11].

III. Fast minimization and sorting algorithms

The vector $\widehat{u}$ is a minimizer of $J(\cdot,f)$ in (2) if and only if $\nabla J(\widehat{u},f)=0$ which is equivalent to $\nabla\Psi(\widehat{u},f)=-\beta\nabla\Phi(\widehat{u})$. By (3) this can be rewritten as

$$\big(\psi'(\widehat{u}[i]-f[i])\big)_{i=1}^{n}=-\beta G^{\mathrm{T}}\big(\varphi'\left((G\widehat{u})[j]\right)\big)_{j=1}^{r}.$$

With $\xi:=(\psi')^{-1}(\cdot,\alpha_1)$ as in (4) and since ξ is odd we obtain

$$\widehat{u}=f-\xi\big(\beta\, G^{\mathrm{T}}\varphi'(G\widehat{u})\big). \tag{9}$$

Here $\varphi'(G\widehat{u}):=\big(\varphi'\left((G\widehat{u})[j]\right)\big)_{j=1}^{r}$ and ξ is applied componentwise. This is a fixed point equation for $\widehat{u}$ which gives rise to the following *fixed point algorithm* to compute $\widehat{u}$:

Algorithm 1 Minimization Algorithm

```
**Initialization:** $u^{(0)}=f$, stopping parameter $\varepsilon$
For $r=1,\dots$ compute until $\|\nabla J\|_\infty\le\varepsilon$
    $u^{(r)}=f-\xi\big(\beta\, G^{\mathrm{T}}\varphi'(Gu^{(r-1)})\big)$
```

As stopping criterion we propose $\|\nabla J\|_\infty\le 10^{-6}$. In all experiments with images of various content and size we realized that the required precision was reached in general within less than 35 iterations. The efficiency of the algorithm relies on two clues:

- By Theorem 2 the vector $u^{(0)}=f$ is very close to the fixed point $\widehat{u}$ and is therefore a good starting point.
- The functions φ' and ξ appearing in the algorithm are given explicitly, see Table I.

By the following corollary the sequence of iterates $\{u^{(r)}\}_{r\in\mathbb{N}}$ is bounded if $\beta<\frac{1}{4}$. Consequently, it has a convergent subsequence. Moreover, for appropriately chosen α_1 all iterates fulfill again the important property $|f[i]-u^{(r)}[i]|<0.5$, $i\in\mathbb{I}_n$ such that the original ordering of the pixels in f is still pertinent in $u^{(r)}$.

Corollary 1. (Distance of $u^{(r)}$ from f: upper bound) *Let ψ and φ fulfill H1 and let $\beta<\frac{1}{4}$. Then, for any $f\in\mathbb{R}^n$, all iterates $u^{(r)}$ generated by Algorithm 1 satisfy*

$$\|u^{(r)}-f\|_\infty\le(\psi')^{-1}\big(4\beta,\alpha_1\big)=\xi\big(4\beta,\alpha_1\big).$$

Proof: By H1 we can estimate

$$\|u^{(r)}-f\|_\infty\le(\psi')^{-1}\big(\beta\|G^{\mathrm{T}}\varphi'(Gu^{(r-1)})\|_\infty\big).$$

Using $|\varphi'(t)|\le 1$ and the sparsity of G^{T} we obtain $\|G^{\mathrm{T}}\varphi'(Gu^{(r-1)})\|_\infty\le 4$ and since $(\psi')^{-1}$ is increasing on $[-1,1]$ for $\beta<\frac{1}{4}$ finally

$$\|u^{(r)}-f\|_\infty\le(\psi')^{-1}\big(4\beta,\alpha_1\big).$$

■

The following theorem provides a convergence result for our fixed point algorithm.

Theorem 3. (Convergence of fixed point algorithm) *Let ψ and φ fulfill H1. Let $\alpha_1,\alpha_2>0$ and $\beta<\frac{1}{4}$ be chosen such that*

$$8\beta\,\xi'(4\beta,\alpha_1)\,\varphi''(0,\alpha_2)<1. \tag{10}$$

Then, for any $f\in\mathbb{R}^n$, the sequence $\{u^{(r)}\}_r$ generated by Algorithm 1 converges to the minimizer $\widehat{u}$ of $J(\cdot,f)$.

Proof: Let $T(u):=f-\xi\big(\beta\, G^{\mathrm{T}}\varphi'(Gu^{(r-1)})\big)$. By Ostrowski's theorem [12] it is enough to prove that the Jacobian matrix $\nabla T(u)$ becomes smaller than 1 in some norm on $\mathbb{R}^n$ for all $u\in\mathbb{R}^n$. Since

$$\nabla T(u)=\beta\operatorname{diag}\big(\xi'(\beta G^{\mathrm{T}}\varphi'(Gu))\big)\,G^{\mathrm{T}}\operatorname{diag}\big(\varphi''\left(Gu\right)\big)\,G$$

we obtain

$$\begin{aligned}\|\nabla T(u)\|_2\le\beta\,&\|\operatorname{diag}\big(\xi'(\beta G^{\mathrm{T}}\varphi'(Gu)\big)\|_2\\ &\|G^{\mathrm{T}}\|_2\,\|\operatorname{diag}\big(\varphi''(Gu)\big)\|_2\,\|G\|_2.\end{aligned}$$

Since φ'' is monotone decreasing on $[0,+\infty)$ we get $\|\operatorname{diag}\big(\varphi''(Gu)\big)\|_2\le\varphi''(0)$. Further, we have by the definition of G that $\|G^{\mathrm{T}}\|_2\,\|G\|_2=\|G^{\mathrm{T}}G\|_2<8$. Note that $G^{\mathrm{T}}G$ is a discrete Laplacian with Neumann boundary conditions and that the bound is sharp in the sense that $\|G^{\mathrm{T}}G\|_2$ approaches 8 as $n\to\infty$.

It remains to estimate $\xi'\left(\beta G^{\mathrm{T}}\varphi'(Gu)\right)$. Regarding that $|\varphi'(t)|\le 1$ for all $t\in\mathbb{R}$ we conclude $\|G^{\mathrm{T}}\varphi'(Gu)\|_\infty\le\|G\|_1\le 4$. Since ξ' increases on $(0,1)$ by (5) and $4\beta<1$ we obtain finally

$$\|\operatorname{diag}\big(\xi'(\beta G^{\mathrm{T}}\varphi'(Gu)\big)\|_2\le\xi'(4\beta).$$

Multiplying the parts together we obtain the assertion. ■

For $\psi=\varphi$ from Table I the left-hand side of (10) becomes

θ_1	θ_2
$\sqrt{\frac{\alpha_1}{\alpha_2}}\,\frac{8\beta}{\sqrt{(1-(4\beta)^2)^3}}$	$\frac{\alpha_1}{\alpha_2}\,\frac{8\beta}{(1-4\beta)^2}$

.

For $\alpha_1=\alpha_2$ these values are smaller than 1 if $\beta<0.0976$ and $\beta<0.0670$, for θ_1 and θ_2, respectively.

Remark 1. *The upper bound in* (10) *is an overestimate since one has $\varphi''(Gu,\alpha_2)=\varphi''(0,\alpha_2)$ only for constant images u.*

For our ordering purpose we are not really interested in the minimizer $\widehat{u}$ of $J(\cdot,f)$, but want to use the sorting of its entries to get a meaningful ordering of the original image. We observed that typically the pixel ordering obtained after a small number of steps of the minimization algorithm does not change in the subsequent steps except for very few pixels. This fact led us to propose the following efficient ordering algorithm for $R\ll 35$ (e.g., $R=5$):

Algorithm 2 Ordering Algorithm

```
**Initialization:** $u^{(0)}=f$, stopping parameter $R$
1. For $r=1,\dots,R$, using $\xi$ and $\varphi'$ in Tab. I, compute
    $u^{(r)}=f-\xi\big(\beta\, G^{\mathrm{T}}\varphi'(Gu^{(r-1)})\big)$
2. Order the values in $\mathbb{I}_n$ according to the corresponding
   ascending entries of $u^{(R)}$.
```

IV. Numerical Comparison of Sorting Algorithms

In this section we demonstrate that our Ordering Algorithm 2 with $\psi = \varphi = \theta_2$ is actually the best way (in terms of speed and quality) to order pixels in digital images. Note that extensive qualitative comparisons of the variational ordering method (VA) with the (fully iterated) Polak-Ribiére algorithm were done in [1]. These experiments have already shown that VA clearly outperforms other state-of-the-art algorithms as LM [8] and WA [9] concerning quality. Here we want to demonstrate that our new ordering algorithm ensures the same quality, in particular a faithful strict ordering, but is much faster than the previous implementations.

We apply VA with parameters $(\beta, \alpha_1, \alpha_2) = (0.1, 0.05, 0.05)$ in the variants

- VA-PR: with Polak-Ribiére algorithm, function θ_1 and stop if $\|\nabla J\|_\infty < 10^{-6}$ but at most 35 iterations as proposed in [1],
- VA-$\theta_k(R)$: with fixed point algorithm with function θ_k, $k \in \{1, 2\}$ and R iterations.

We want to mention that the estimate in Theorem 3 is too restrictive (see Remark 1) and we have chosen β slightly larger which still provides a convergent iteration scheme. Further we use the notation LM(K) and WA(K) for the corresponding algorithms with $K - 1$ filter applications. Recall that $K = 6$ for LM and $K = 9$ for WA were recommended by the authors.

Remark 2. (Filtering versus sorting)
The above algorithms contain a filtering and a sorting step which behave quite differently:
- LM and WA: Both algorithms apply $K - 1$ simple linear filters which is cheap. For example, if $n = N^2$ is the number of image pixels, then $24n, 44n, 64n$ (mainly) additions are necessary for the LM-ϕ filtering procedure in the cases $K = 4, 5, 6$, respectively. Note that this filtering can be also done in a cheaper way using the ψ-filters employed in [8] for the theoretical study of the LM algorithm. The resulting K images must be lexicographically ordered, see (1). This can be done in $\mathcal{O}(n \log n)$, but the concrete factor depends on K and the image content. In our numerical experiments we have used the `sortrows` Matlab function which calls for $K \geq 4$ a C program. In our numerical examples with $K = 6$ this sorting procedure was three to twelve times (increasing with increasing number of pixels) slower than the filtering procedure. Finally, we mention that larger images, e.g., of size 5616×3744 taken by usual commercial cameras, cannot be handled by `sortrows`. Here a more sophisticated sequential sorting implementation in a better adapted programming language may be used with storage requirement $2n$. However, the speed relation between linear filtering and lexicographical sorting will be kept.
- VA: The nonlinear filtering in the VA procedure is more demanding than the above linear one. However, for $\varphi = \psi = \theta_2$ we have (up to absolute values) only to compute additions and multiplications. In summary, we have to perform $13n$ additions or multiplications in each iteration step. Indeed our numerical experiments have confirmed that our filtering behaves as $\mathcal{O}(n)$. The subsequent sorting procedure `sort` of one image which requires $n \log n$ operations is faster than the filtering step, in our examples with $R = 5$ nearly 4 times.

We summarize: for LM(K) and WA(K), the lexicographical sorting of K images is more time consuming than the simple linear filtering. In our ordering algorithm, the nonlinear filtering requires more time than the sorting step. The running time for our filtering is linear in the number of pixels n and is cheaper for θ_2 than for θ_1.

All algorithms are implemented in Matlab2012a and executed on a computer with an Intel Core i7-870 Processor (8M Cache, 2.93 GHz) and 8 GB physical memory, 64 Bit Linux. The tests are performed for four groups of digital 8-bit images of increasing size $N \times N$, where $N = 256, 512, 1024, 2048$, presented in Fig. 2. The results in all tables give *the means over 100 runs for each image* of all tested algorithms.

We present two numerical experiments:

1. Ordering of natural images: The results for LM(6), WA(9) and the variational approaches are reported in Tab. II. Here `Fail` gives the percentage of image pixels which cannot be totally ordered. The fast VA algorithm with θ_2 and 5 iterations VA-θ_2(5) clearly outperform the LM(6) and WA(9) algorithms both with respect to `Fail` and computational time. The different VA algorithms show a similar `Fail`. However, compared to the Polak-Ribiére algorithm with θ_1 suggested in [1], VA-θ_2(5) reduces the running time by a factor of 50 to 30. Further, our fixed point algorithm needs 20 % less computation time for VA-θ_2 than for VA-θ_1.

Tab. III shows the fail rate of LM and VA-θ_2 for numbers $K-1$ and R of filter applications. It can be seen that after $R = 2$ applications of our nonlinear filter, the average percentage of pixels which cannot be sorted is nearly the same as for LM(6). However, for $R = 5$ steps of our nonlinear filter, nearly all pixels are faithfully sorted.

Fig. 3 depicts the filtering time, the sorting time and the overall running time of LM(K) and VA-$\theta_2(R)$ for $R = K - 1 \in \{1, \ldots, 5\}$. The plots on the left (resp., on the right) give the means over 100 runs for all images in Fig. 2 of size 512×512 (resp., of size 1024×1024). The obtained curves are in agreement with Remark 2.

2. Histogram equalization inversion: In [8], histogram equalization inversion was shown to be a relevant way to evaluate up to what degree a sorting algorithm is meaningful. First the original 8-bit image f with histogram h_f is mapped to an 8-bit image g with a uniform histogram (up to rounding errors). This requires the first application of an ordering algorithm. Then g is transformed to an 8-bit image $\tilde{f}$ with histogram h_f which requires a second time an ordering algorithm. Tab. IV shows the PSNR $20 \log_{10}(255 M \cdot N / \|f - \tilde{f}\|_2)$, the percentage of pixels `Fail%` which cannot be faithfully ordered averaged over the two applied ordering procedures and the computational time of the whole histogram equalization inversion process. Since VA-PR and VA-$\theta_1(R)$ give qualitatively, in terms of PSNR and FAIL, the same results as VA-$\theta_2(R)$ but VA-$\theta_2(R)$ is faster, we consider only VA-$\theta_2(R)$, $R \in \{5, 35\}$. The VA-algorithms outperform LM(6) and WA(9) wrt PSNR and FAIL. Moreover, VA-$\theta_2(5)$ is the

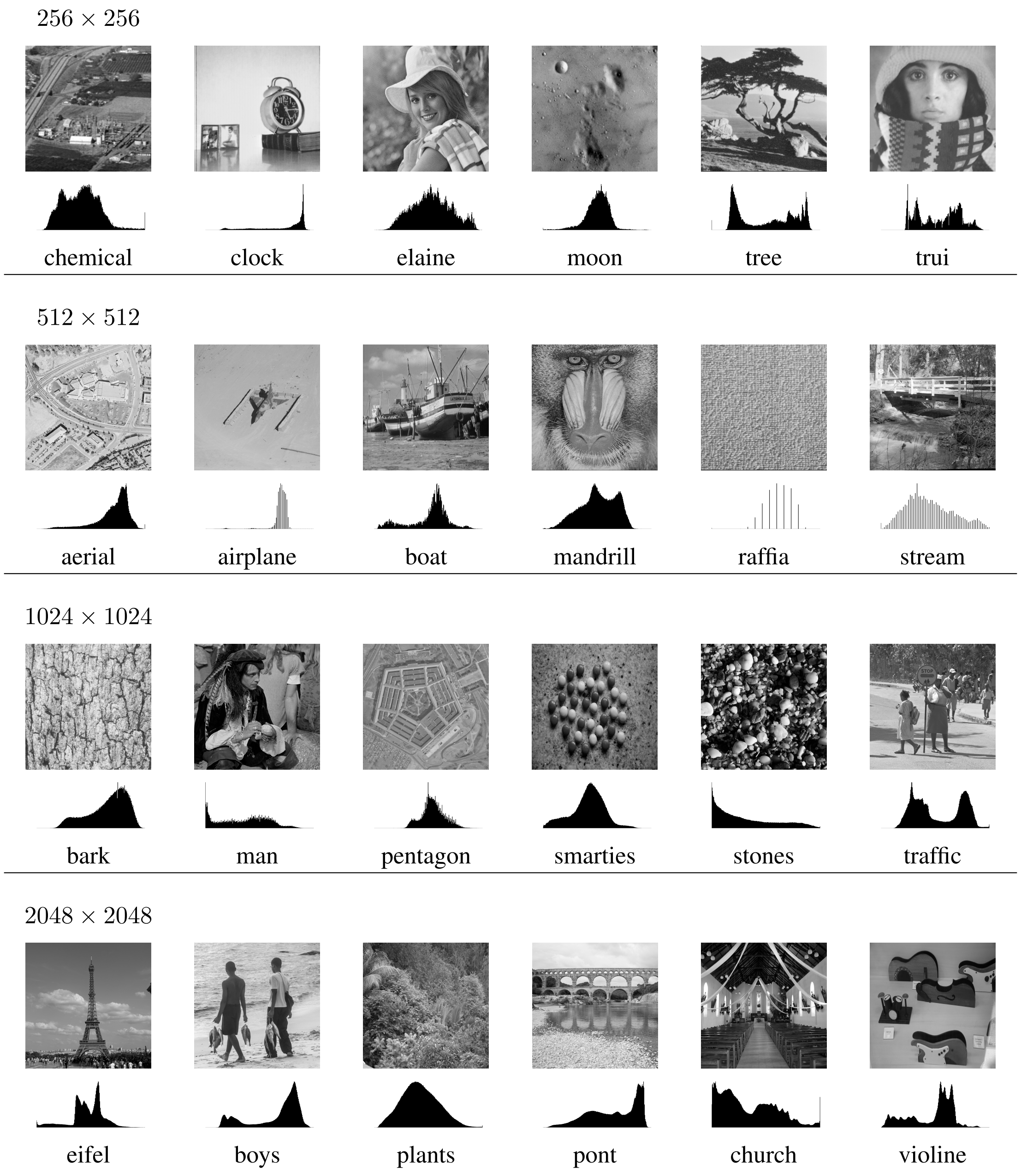

Fig. 2. All 24 digital 8-bit images with their histograms used for our comparison. The size of the images ranges from 256×256 in the first row to 2048×2048 in the fourth row.

fastest algorithm.

The quality of our VA algorithms is emphasized by Fig. 4 which shows three difference images of the original image f and the images $\tilde{f}$ obtained after the histogram equalization inversion. The first row presents the original image `trui` and zooms of `stones` and `church`. The second and third rows show the results $f - \tilde{f}$ obtained by the LM and WA ordering, respectively. The fourth and the fifth rows depict the difference images $f - \tilde{f}$ corresponding to VA-$\theta_2(35)$ and VA-$\theta_2(5)$, respectively. Both VA methods are able to reconstruct the original image more precisely than their competitors in particular in the vicinity of edges.

V. Conclusions and Future Work

In this paper we have proposed a fast algorithm to order strictly and reasonably the pixels in digital (integer valued) natural images. Our algorithm outperforms by far the state-of-the-art algorithms in speed and in quality to order the pixels. Thus it provides a background for any exact histogram specification based application.

We have applied this algorithm for hue and range preserving enhancement of color images in [2] where we also handled large size images. Given an RGB image $w = (w_r, w_g, w_b)$ the rough idea consists of the following two steps:

(i) appropriate histogram specification of the intensity image $f := \frac{1}{3}(w_r + w_g + w_b)$ using our proposed sorting algorithm which yields an enhanced intensity image $\hat{f}$.

	Fail %						Computation Time					
			Variational Method						Variational Method			
method	LM(6)	WA(9)	PR	$\theta_1(35)$	$\theta_2(35)$	$\theta_2(5)$	LM(6)	WA(9)	PR	$\theta_1(35)$	$\theta_2(35)$	$\theta_2(5)$
256×256												
chemical	0.01	0.06	0.00	0.00	0.00	0.00	0.03	0.04	0.59	0.08	0.06	0.01
clock	4.82	4.52	0.00	0.00	0.00	0.00	0.03	0.04	0.27	0.08	0.06	0.01
elaine	0.01	0.02	0.00	0.00	0.00	0.00	0.03	0.03	0.61	0.08	0.07	0.01
moon	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.04	0.61	0.08	0.06	0.01
tree	0.20	0.19	0.00	0.00	0.00	0.00	0.03	0.04	0.59	0.08	0.06	0.01
trui	0.03	0.00	0.00	0.00	0.00	0.00	0.03	0.04	0.30	0.08	0.06	0.01
means	0.84	0.80	0.00	0.00	0.00	0.00	0.03	0.04	0.50	0.08	0.06	0.01
512×512												
aerial	0.04	0.00	0.00	0.00	0.00	0.00	0.13	0.16	2.28	0.45	0.29	0.05
airplane	18.58	17.70	0.00	0.00	0.00	0.00	0.16	0.31	1.16	0.44	0.29	0.05
boat	0.00	0.00	0.00	0.00	0.00	0.00	0.12	0.15	1.40	0.45	0.29	0.05
mandrill	0.00	0.00	0.00	0.00	0.00	0.00	0.12	0.14	2.25	0.43	0.29	0.05
raffia	46.64	16.05	0.00	0.00	0.00	0.00	0.20	0.35	1.04	0.44	0.29	0.05
stream	0.65	0.75	0.00	0.00	0.00	0.14	0.13	0.19	1.03	0.44	0.29	0.05
means	10.98	5.75	0.00	0.00	0.00	0.02	0.14	0.22	1.53	0.44	0.29	0.05
1024×1024												
bark	0.00	0.00	0.00	0.00	0.00	0.00	0.64	0.80	10.16	1.79	1.33	0.25
man	0.43	0.68	0.00	0.00	0.00	0.00	0.61	0.82	4.47	1.62	1.22	0.24
pentagon	0.00	0.00	0.00	0.00	0.00	0.00	0.66	0.87	9.48	1.80	1.32	0.22
smarties	0.55	0.05	0.00	0.00	0.00	0.00	0.68	0.90	9.43	1.82	1.46	0.24
stones	2.25	1.39	0.00	0.00	0.00	0.09	0.66	0.90	4.76	1.78	1.32	0.22
traffic	0.69	0.07	0.00	0.00	0.00	0.00	0.65	0.88	4.96	1.80	1.44	0.24
means	0.65	0.36	0.00	0.00	0.00	0.01	0.65	0.86	7.21	1.77	1.34	0.24
2048×2048												
eifel	3.54	0.37	0.00	0.00	0.00	0.01	4.24	6.56	22.29	8.08	6.58	1.23
boys	0.33	0.00	0.00	0.00	0.00	0.00	3.84	5.45	42.64	8.06	6.60	1.20
plants	0.03	0.00	0.00	0.00	0.00	0.00	3.74	4.73	21.61	8.08	6.62	1.20
pont	9.98	5.77	0.00	0.00	0.00	0.00	3.98	5.67	43.15	8.08	6.63	1.20
church	0.96	0.78	0.13	0.05	0.07	0.30	3.62	5.18	20.74	8.08	6.60	1.21
violine	1.83	0.23	0.00	0.00	0.00	0.00	3.76	5.97	43.25	8.07	6.59	1.19
means	2.78	1.19	0.02	0.01	0.01	0.05	3.86	5.59	32.28	8.07	6.60	1.20

TABLE II
COMPARISON OF DIFFERENT ORDERING METHODS FOR THE IMAGES IN FIG. 2.

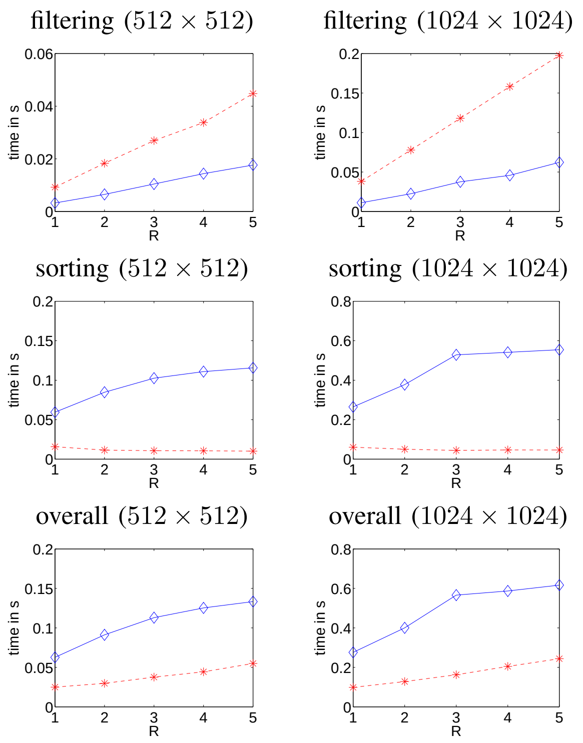

Fig. 3. Comparison of filtering, sorting and overall computational time of LM(K) (solid $\diamond$) and VA-$\theta_2(R)$ (dashed $*$) for $R = K - 1 = 1, \dots, 5$ for the images in Fig. 2 of size 512×512 (left) and 1024×1024 (right).

(ii) application of an affine transform to each pixel of the RGB image w such that the resulting image $\hat{w} = (\hat{w}_r, \hat{w}_g, \hat{w}_b)$ has the desired intensity $\hat{f}$ and is in the range $[0, 255]$. The images w and $\hat{w}$ have the same hue.

REFERENCES

[1] M. Nikolova, Y.-W. Wen, and R. Chan, "Exact histogram specification for digital images using a variational approach", *Journal of Mathematical Imaging and Vision*, vol. 46, no. 3, pp. 309–325, July 2013.

[2] M. Nikolova and G. Steidl, "Hue and range preserving color image enhancement based on fast histogram specification. New algorithms, theory and applications", *IEEE Transactions on Image Processing*, vol. 23, no. 9, pp. 4087–4100, Sep. 2014.

[3] R. Gonzalez and R. Woods, *Digital Image Processing*, Addison-Wesley, 1993.

[4] V. Caselles, J. L. Lisani, J. M. Morel, and G. Sapiro, "Shape preserving local histogram modification", *IEEE Transactions on Image Processing*, vol. 8, no. 2, pp. 220–229, Feb. 1999.

[5] D. Sen and P. Sankar, "Automatic exact histogram specification for contrast enhancement and visual system based quantitative evaluation", *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1211–1220, May 2011.

[6] N. Bassiou and C. Kotropoulos, "Color image histogram equalization by absolute discounting back-off", *Computer Vision and Image Understanding*, vol. 107, 2007.

[7] E. L. Hall, "Almost uniform distributions for computer image enhancement", *IEEE Transactions on Computers*, vol. C-23, no. 2, pp. 207–208, Feb. 1974.

[8] D. Coltuc, P. Bolon, and J.-M. Chassery, "Exact histogram specification", *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1143–1152, 2006.

	Fail %									
method	LM(2)	LM(3)	LM(4)	LM(5)	LM(6)	$\theta_2(1)$	$\theta_2(2)$	$\theta_2(3)$	$\theta_2(4)$	$\theta_2(5)$
256×256										
chemical	68.93	6.40	0.19	0.03	0.01	2.97	0.07	0.00	0.00	0.00
clock	74.55	43.56	23.61	10.76	4.82	47.51	8.69	0.47	0.02	0.00
Elaine	75.02	12.00	0.56	0.03	0.01	7.78	0.06	0.00	0.00	0.00
moon	76.65	8.57	0.20	0.00	0.00	4.60	0.00	0.00	0.00	0.00
tree	67.89	17.65	4.74	0.91	0.20	19.51	1.12	0.00	0.00	0.00
trui	83.46	32.82	4.14	0.31	0.03	31.72	0.07	0.00	0.00	0.00
means	74.42	20.16	5.57	2.01	0.84	19.02	1.67	0.08	0.00	0.00
512×512										
aerial	85.69	34.97	5.10	0.37	0.04	23.05	0.02	0.00	0.00	0.00
airplane	96.38	86.27	67.71	38.24	18.58	91.07	38.06	0.19	0.00	0.00
boat	87.59	23.76	0.92	0.02	0.00	15.67	0.00	0.00	0.00	0.00
mandrill	81.17	7.36	0.12	0.00	0.00	2.59	0.00	0.00	0.00	0.00
raffia	99.91	99.22	95.08	77.35	46.64	99.51	29.02	0.15	0.01	0.00
stream	95.35	53.66	13.28	2.06	0.65	56.38	1.53	0.32	0.21	0.14
means	91.02	50.87	30.37	19.67	10.98	48.04	11.44	0.11	0.04	0.02
1024×1024										
bark	96.19	31.40	0.60	0.01	0.00	5.62	0.00	0.00	0.00	0.00
man	95.82	48.63	6.92	1.29	0.43	28.85	0.91	0.09	0.00	0.00
Pentagon	96.42	46.89	2.21	0.03	0.00	23.90	0.00	0.00	0.00	0.00
smarties	97.51	78.66	29.81	4.09	0.55	49.21	0.16	0.02	0.01	0.00
stones	96.09	71.57	30.10	7.24	2.25	49.18	2.32	0.54	0.21	0.09
traffic	96.41	68.06	25.57	4.41	0.69	44.93	0.48	0.00	0.00	0.00
means	96.41	57.54	15.87	2.85	0.65	33.61	0.64	0.11	0.04	0.01
2048×2048										
eifel	98.89	87.50	65.23	23.36	3.54	81.26	0.32	0.05	0.02	0.01
boys	99.22	87.48	41.00	5.02	0.33	69.22	0.00	0.00	0.00	0.00
plants	98.27	68.04	12.72	0.56	0.03	31.92	0.01	0.00	0.00	0.00
pont	98.86	78.36	30.59	18.09	9.98	44.87	10.54	0.02	0.00	0.00
church	98.62	82.32	30.77	3.71	0.96	66.01	1.07	0.48	0.36	0.30
violine	99.41	92.91	61.18	15.52	1.83	87.59	0.12	0.00	0.00	0.00
means	98.88	82.77	40.25	11.04	2.78	63.48	2.01	0.09	0.06	0.05

TABLE III
COMPARISON OF LM(K) WITH VA-$\theta_2(R)$ WITH RESPECT TO FAIL FOR AN INCREASING NUMBER $R = K - 1$ OF FILTER APPLICATIONS.

[9] Y. Wan and D. Shi, "Joint exact histogram specification and image enhancement through the wavelet transform", *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2245–2250, 2007.

[10] M. Nikolova, "A fast algorithm for exact histogram specification. simple extension to colour images", in *Scale Space and Variational Methods in Computer Vision*. Lecture Notes in Computer Science 7893, Springer, 2013, pp. 174–185.

[11] F. Baus, M. Nikolova, and G. Steidl, "Fully smoothed ℓ_1-TV models: Bounds for the minimizers and parameter choice", *Journal of Mathematical Imaging and Vision*, vol. 48 no. 2 pp. 295–307, 2014.

[12] J. Ortega and W. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

	PSNR				Fail %				Computation Time			
			VA				VA				VA	
method	LM(6)	WA(9)	$\theta_2(35)$	$\theta_2(5)$	LM(6)	WA(9)	$\theta_2(35)$	$\theta_2(5)$	LM	WA	$\theta_2(35)$	$\theta_2(5)$
256×256												
chemical	49.34	48.90	49.67	49.67	0.03	0.10	0.00	0.00	0.06	0.07	0.13	0.03
clock	51.69	51.56	51.78	51.78	0.87	2.41	0.00	0.02	0.06	0.08	0.14	0.02
elaine	49.51	49.66	49.90	49.90	0.00	0.02	0.00	0.00	0.06	0.07	0.13	0.03
moon	47.36	46.50	47.82	47.77	0.06	0.11	0.00	0.00	0.05	0.07	0.13	0.03
tree	51.94	51.84	52.01	52.01	0.03	0.18	0.00	0.00	0.05	0.07	0.12	0.03
trui	52.70	52.51	52.86	52.85	0.04	0.06	0.00	0.00	0.05	0.07	0.13	0.03
means	50.42	50.16	50.67	50.67	0.17	0.48	0.00	0.00	0.05	0.07	0.13	0.03
512×512												
aerial	48.36	48.06	50.05	50.05	0.00	0.02	0.00	0.00	0.25	0.33	0.57	0.11
airplane	46.74	46.26	47.25	47.25	2.68	8.96	0.00	0.01	0.27	0.49	0.56	0.10
boat	49.51	49.58	49.89	49.89	0.07	0.09	0.00	0.00	0.24	0.32	0.57	0.11
mandrill	48.27	49.47	49.75	49.76	0.00	0.00	0.00	0.00	0.23	0.30	0.56	0.10
raffia	41.12	41.12	41.12	41.12	6.85	8.18	0.00	0.00	0.32	0.54	0.57	0.11
stream	44.75	45.00	45.07	45.08	0.40	0.71	0.00	0.13	0.24	0.36	0.56	0.10
means	46.46	46.58	47.19	47.19	1.67	2.99	0.00	0.02	0.26	0.39	0.57	0.11
1024×1024												
bark	51.28	51.17	51.30	51.30	0.00	0.01	0.00	0.00	1.29	1.60	2.60	0.48
man	49.22	49.18	49.44	49.44	0.19	0.40	0.00	0.00	1.20	1.59	2.55	0.46
pentagon	50.69	50.62	51.35	51.35	0.01	0.01	0.00	0.00	1.29	1.66	2.53	0.48
smarties	51.47	51.09	51.64	51.60	0.63	0.91	0.00	0.03	1.34	1.77	2.55	0.49
stones	51.28	50.95	51.60	51.60	0.97	1.22	0.00	0.04	1.31	1.78	2.52	0.44
traffic	50.73	50.58	51.01	51.00	0.25	0.30	0.00	0.01	1.31	1.73	2.53	0.47
means	50.78	50.60	51.06	51.05	0.34	0.48	0.00	0.01	1.29	1.69	2.55	0.47
2048×2048												
eifel	48.61	48.48	48.74	48.73	0.88	0.97	0.00	0.05	8.17	12.36	13.15	2.41
boys	51.63	51.43	51.71	51.71	0.09	0.13	0.00	0.00	7.45	10.44	13.15	2.38
plants	48.96	48.86	49.52	49.52	0.18	0.24	0.00	0.02	7.35	9.29	13.15	2.39
pont	51.48	51.30	51.55	51.51	1.42	3.50	0.00	0.09	7.78	10.90	13.24	2.39
church	50.94	50.80	51.28	51.27	1.23	1.73	0.06	0.28	7.16	10.23	13.16	2.40
violine	51.66	51.34	51.85	51.75	1.22	1.59	0.10	0.22	7.25	11.29	13.12	2.37
means	50.55	50.37	50.78	50.75	0.84	1.36	0.03	0.11	7.53	10.75	13.16	2.39

TABLE IV
COMPARISON OF HISTOGRAM EQUALIZATION INVERSION ALGORITHMS.

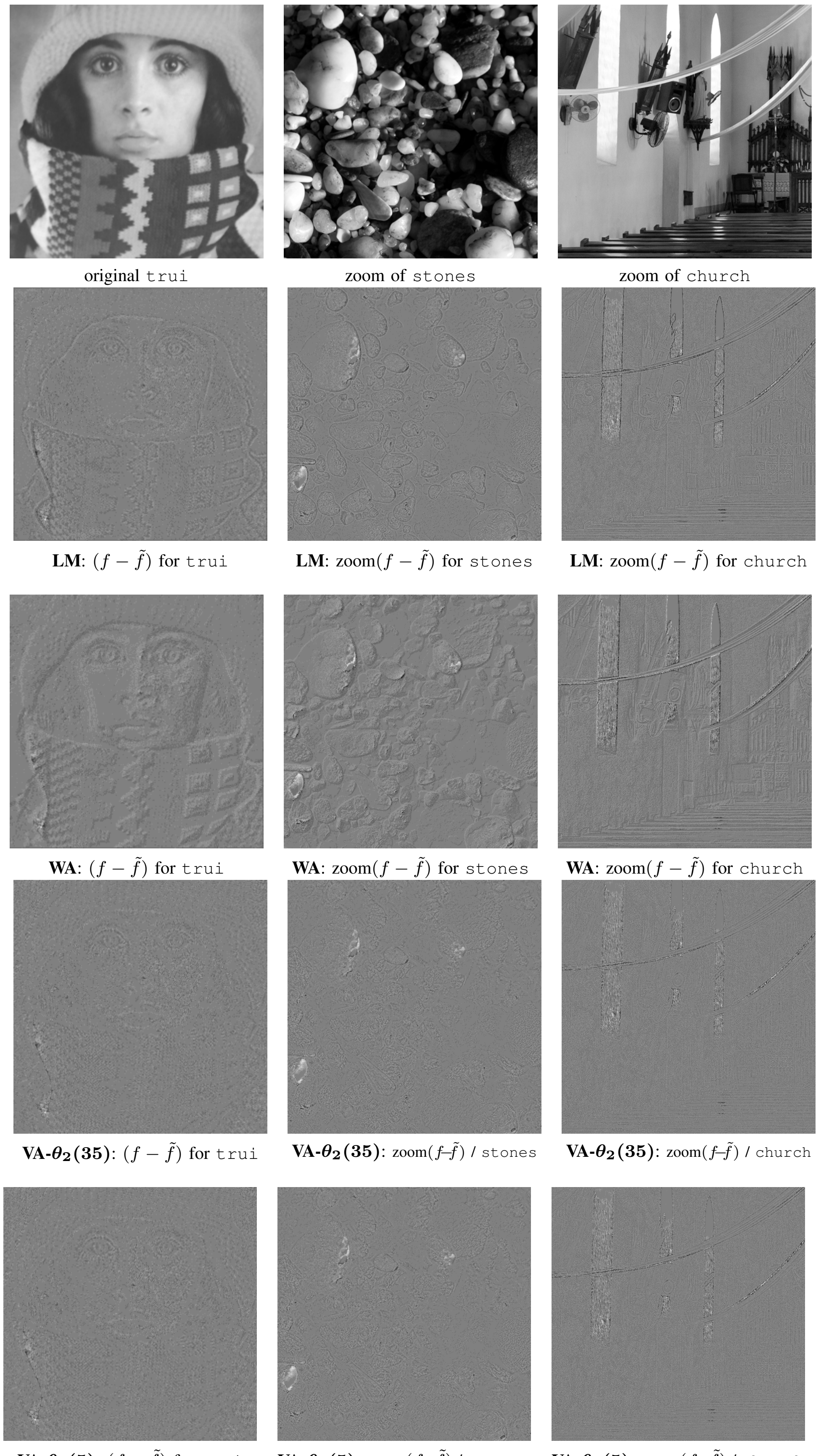
original trui
zoom of stones
zoom of church
LM: $(f - \tilde{f})$ for trui
LM: zoom$(f - \tilde{f})$ for stones
LM: zoom$(f - \tilde{f})$ for church
WA: $(f - \tilde{f})$ for trui
WA: zoom$(f - \tilde{f})$ for stones
WA: zoom$(f - \tilde{f})$ for church
VA-$\theta_2(35)$: $(f - \tilde{f})$ for trui
VA-$\theta_2(35)$: zoom$(f–\tilde{f})$ / stones
VA-$\theta_2(35)$: zoom$(f–\tilde{f})$ / church
VA-$\theta_2(5)$: $(f - \tilde{f})$ for trui
VA-$\theta_2(5)$: zoom$(f–\tilde{f})$ / stones
VA-$\theta_2(5)$: zoom$(f–\tilde{f})$ / church

Fig. 4. Comparison of ordering methods for histogram equalization inversion. First row: true images or parts of them. The following rows show the difference images between the original one and those obtained after histogram equalization inversion. Top to down: LM(6), WA(9), VA-$\theta_2(35)$, VA-$\theta_2(5)$. The variational methods (VA) contain much less errors than those achieved by LM(6) and WA(9). Moreover there is no visual difference between VA with 35 iterations and its faster variant with only 5 iterations.