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 is based on the ordering procedure relying 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. In particular the original image is a pertinent starting point for the iterations and all functions involved in the algorithm were shown to admit a simple explicit form which is quite exceptional. Only few iteration steps of this algorithm provide images whose pixels can be ordered in a strict and faithful way. Equivalently, our algorithm can be seen as iterative nonlinear filtering. Numerical experiments confirm that our algorithm outperforms by far its main competitors. Our minimization algorithm and ordering approach provide the background for any exact histogram specification based application.

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 invisible watermarking, image normalization and enhancement, object recognition [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).

For a uniform target histogram we speak about histogram equalization (HE).

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

$$i \prec j \text{ if } \kappa_s[i] < \kappa_s[j] \text{ for some } s \in \mathbb{I}_K$$

and

$$\kappa_k[i] = \kappa_k[j] \text{ for all } 1 \leq k < s. \quad (1)$$

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-Ribière algorithm and the whole ordering algorithm was slower than LM and WA.

The main contribution of this paper consists in providing a simple fixed point algorithm that attains the minimizer with remarkable speed and precision. Convergence and parameter selection are discussed based on theoretical results. Our minimization scheme amounts to apply a particular nonlinear filter.
In practice, only few iterations with this nonlinear filter are sufficient to provide the information needed for a meaningful strict ordering of the values. In contrast to LM and WA our algorithm requires just a single ordering of one image and not the lexicographical ordering of $K$ images needed for the LM and WA method. Our new algorithm provides the important speedup technique which makes VA competitive with LM and VA. Numerical tests confirm that the VA approach along with the new fixed point algorithm outperforms by far all other relevant ordering methods in terms of quality and speed. Some preliminary results of this paper were published in the conference paper [10].

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., invisible watermarking, image enhancement among others. We have used our algorithm successfully for hue and range preserving HS based color image enhancement in [2].

The outline of the paper is as follows: In Section II we review the special variational approach in [1] and some of its properties proven in [11]. Then, in Section III, we propose a simple fixed point algorithm to find a minimizer of our functional. The reasons for its efficiency and its effectiveness are explained. Section IV contains 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. We will see that only few iterations of our algorithm are necessary to obtain promising ordering results. Finally, conclusions are given in Section V.

II. THE FULLY SMOOTHED $\ell_1$–TV MODEL

Let $D_N$ denote the forward difference matrix

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

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 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

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

with

$$\Psi(u, f) := \sum_{i \in \mathbb{Z}_n} \psi(u[i] - f[i]),$$

$$\Phi(u) := \sum_{j \in \mathbb{Z}_r} \varphi((Gu)[j]).$$ (3)
Here \((Gu)[j]\) denotes the \(j\)th 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 \(x\) and \(y\) 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 \(\theta(\cdot, \cdot) : \mathbb{R} \to \mathbb{R}, \alpha > 0\), satisfying the requirements in assumptions H1 and H2 described next. The rationale for these choices was extensively discussed in [1]. For simplicity, the parameters \(\alpha_1\) and \(\alpha_2\) 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 \(C^2(\mathbb{R})\) 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 \(t \mapsto \theta(t, \alpha)\) is strictly decreasing on \((0, +\infty)\) with

\[
\lim_{\alpha \to 0} \theta'(t, \alpha) = 1 \quad \text{and} \quad \lim_{\alpha \to \infty} \theta'(t, \alpha) = 0.
\]

Under these assumptions \(\psi\) and \(\varphi\) are smooth approximations of the absolute value function. Hence the functional \(J(\cdot, f)\) amounts to a fully smoothed \(l_1\)-TV model.

\(\hat{\xi}\)From H1 it follows that \(\theta'(t, \alpha)\) is odd and has an inverse function

\[
\xi(t, \alpha) := (\theta')^{-1}(t, \alpha).
\]

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 \(\xi\) is differentiable and

\[
\xi'(t, \alpha) = \frac{1}{\theta''(\xi(t))} > 0.
\]

So \(t \mapsto \xi'(t, \alpha)\) is also increasing on \((0, 1)\). There are many possible choices of functions \(\theta\) meeting H1 and H2, see [1].

**Example 1.** In our numerical tests we use the functions \(\theta = \psi = \varphi\) with \(\alpha_1 = \alpha_2 = \alpha\) given in the following table:

<table>
<thead>
<tr>
<th>(\alpha)</th>
<th>(\theta)</th>
<th>(\theta')</th>
<th>(\theta'')</th>
<th>(\xi)</th>
<th>(\xi')</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\sqrt{1 + \alpha}^2 + \alpha)</td>
<td>(\frac{1}{\sqrt{1 + \alpha}^2 + \alpha})</td>
<td>(\frac{1}{1 + \alpha})</td>
<td>(\frac{\alpha}{(1 + \alpha)^2})</td>
<td>(\frac{1}{\alpha \cdot (1 + \alpha)^2})</td>
<td>(\frac{1}{(1 + \alpha)^3})</td>
</tr>
</tbody>
</table>

**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 \(\hat{u} \in \mathbb{R}^n\). The following 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] guaranties that the entries of the minimizer differ in general pairwise from each other so that \(\hat{u}\) provides an auxiliary information for ordering the pixels of \(f\).

**Theorem 1.** (Strict ordering information)
Let \(\psi\) and \(\varphi\) 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 \(\hat{u}\) of \(J(\cdot, f)\) satisfies

\[
\hat{u}[i] \neq \hat{u}[j], \quad \forall \ i, j \in \mathbb{N}_n, \ i \neq j,
\]

\[
\hat{u}[i] \neq f[i], \quad \forall \ i \in \mathbb{N}_n.
\]

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 \(\mathbb{R}^n\). Therefore, the elements of \(\mathbb{R}^n \setminus \mathbb{K}^n\) are highly exceptional in \(\mathbb{R}^n\).

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

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

\[
\|\hat{u} - f\|_\infty \leq (\psi')^{-1}(4\beta, \alpha_1) = \xi(4\beta, \alpha_1),
\]

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

\[
\|\hat{u} - f\|_\infty \geq \xi(4\beta, \alpha_1) \quad \text{as} \quad \alpha_2 \to 0 \quad (8)
\]

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

The upper bound (7) guarantees in our numerical examples that \(|f[i] - \hat{u}[i]| < 0.5\) for any \(i \in \mathbb{N}_n\). Consequently, if for \(f[i] \in \{0, \ldots, 255\}\), \(i \in \mathbb{N}_n\), the relation \(f[i] < f[j]\) holds true, then also \(\hat{u}[i] < \hat{u}[j]\) 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 \(\|\hat{u} - f\|_\infty \leq 0.0976\) if \(\psi = \varphi = \theta_1\) and \(\|\hat{u} - f\|_\infty \leq 0.0333\) if \(\psi = \varphi = \theta_2\).

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

**III. FAST MINIMIZATION AND SORTING ALGORITHMS**

The function \(\hat{u}\) is a minimizer of \(J(\cdot, f)\) in (2) if and only if \(\nabla J(\hat{u}, f) = 0\) which is equivalent to \(\nabla \Psi(\hat{u}) = -\beta \nabla \Phi(\hat{u})\).

By (3) this can be rewritten as

\[
(\psi'(\hat{u}[i] - f[i]))_{i=1}^{n} = -\beta G^t(\varphi'((G\hat{u})[j]))_{j=1}^{n}.
\]

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

\[
\hat{u} = \beta \xi G^t(\varphi(G\hat{u})).
\]

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

**Algorithm 1 Minimization Algorithm**

**Initialization**: $u^{(0)} = f$, stopping parameter $\varepsilon$

For $r = 1, \ldots$ compute until $\|\nabla J\|_\infty \leq \varepsilon$

$$u^{(r)} = f - \xi(\beta^r \varphi'(Gu^{(r-1)}))$$

As stopping criterion we propose $\|\nabla J\|_\infty \leq 10^{-6}$. 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 $\hat{u}$ and is therefore a good starting point.
- The functions $\varphi'$ and $\xi$ 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 $\alpha_1$ all iterates fulfill again the important inequality $|f[i] - u^{(r)}[i]| \leq 0.5$, $i \in \mathbb{N}$, 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 $\psi$ and $\varphi$ 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

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

**Proof.** By $H1$ we can estimate

$$\|u^{(r)} - f\|_\infty \leq (\psi')^{-1}(\beta \|G^r \varphi'(Gu^{(r-1)})\|_\infty).$$

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

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

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

**Theorem 3.** (Convergence of fixed point algorithm)

Let $\psi$ and $\varphi$ fulfill $H1$. Let $\alpha_1, \alpha_2 > 0$ and $\beta < \frac{1}{4}$ be chosen such that

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

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

**Proof.** Let $T(u) := f - \xi(\beta^r \varphi'(Gu^{(r-1)}))$. 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

$$\nabla T(u) = \beta \text{diag}(\xi'(\beta^r \varphi'(Gu))) G^r \text{diag}(\varphi''(Gu)) G$$

we obtain

$$\|\nabla T(u)\|_2 \leq \beta \|\text{diag}(\xi'(\beta^r \varphi'(Gu))) G^r \|_2 \|\text{diag}(\varphi''(Gu))\|_2.$$

Since $\varphi''$ is monotone decreasing on $[0, +\infty)$ we get

$$\|\text{diag}(\varphi''(Gu))\|_2 \leq \varphi''(0).$$

Further, we have by the definition of $G$ that $\|G^r\|_2 \|G\|_2 = \|G^r G\|_2 < 8$. Note that $G^r G$ is a discrete Laplacian with Neumann boundary conditions and that the bound is sharp in the sense that $\|G^r G\|_2$ approaches 8 as $n \to \infty$.

It remains to estimate $\xi'(\beta^r \varphi'(Gu))$. Regarding that $|\varphi'(t)| \leq 1$ for all $t \in \mathbb{R}$ we conclude $\|\text{diag}(\varphi''(Gu))\|_2 \leq \|G\|_1 \leq 4$. Since $\xi'$ increases on $(0, 1)$ by (5) and $4\beta < 1$ we obtain finally

$$\|\text{diag}(\xi'(\beta^r \varphi'(Gu)))\|_2 \leq \xi'(4\beta).$$

Multiplying the parts together we obtain the assertion.

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

$$\theta_1 = \frac{\alpha_1}{\alpha_2} \sqrt{\frac{1}{1 - (4\beta)^2}}, \quad \theta_2 = \frac{\alpha_1}{\alpha_2} \frac{8\beta}{1 - (4\beta)^2}.$$

For $\alpha_1 = \alpha_2$ these values are smaller than 1 if $\beta < 0.0976$ and $\beta < 0.0670$, for $\theta_1$ and $\theta_2$, respectively.

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

In practice, we are not really interested in the minimizer $\hat{u}$ of $J(\cdot, f)$, but want to use the sorting of its components to get a meaningful ordering of the original image. Here we observed that the ordering of pixels obtained after a small number of steps of the minimization algorithm does not change in the subsequent steps except of very few pixels. This fact led us to propose the following efficient ordering algorithm for $R \ll 35$:

**Algorithm 2 Ordering Algorithm**

**Initialization**: $u^{(0)} = f$, stopping parameter $R$

1. For $r = 1, \ldots, R$ compute

$$u^{(r)} = f - \xi(\beta^r \varphi'(Gu^{(r-1)}))$$

2. Order the values in $\mathbb{N}$ according to the corresponding ascending entries of $u^{(R)}$.

Step 1 is an iterative nonlinear filtering procedure.

**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-Ribière 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 LM with $K = 6$, WA with $K = 9$ and VA with parameters $(\beta, \alpha_1, \alpha_2) = (0.1, 0.05, 0.05)$ in the variants
- VA-PR: with Polak-Ribiére algorithm, function $\theta_1$ and stop if $\|\nabla J\|_\infty < 10^{-6}$ but at most 35 iterations as proposed in [1],
- VA-\(k\theta_k(R):\) with fixed point algorithm with function $\theta_k$,
  $k \in \{1, 2\}$ and $R$ iterations.

Recall that $K = 6$ for LM and $K = 9$ for WA were recommended by the authors. Further, we want to mention that the estimate in Theorem 3 is too restrictive (see Remark 1) and we have chosen $\beta$ slightly larger which still provides a convergent iteration scheme.

**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-$\phi$ filtering procedure in the cases $K = 4, 5, 6$, respectively. Note that this filtering can be also done in a cheaper way using the $\psi$-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 $O(n \log n)$ but the concrete factor depends on $K$ and on 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 $5616 \times 3744$ 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 $O(n)$. The subsequent sorting procedure sort of one image which requires $n \log n$ is faster than the filtering step, in our examples with $R = 5$ nearly 4 times.

We summarize our findings for images up to size $2048 \times 2048$: For LM and WA the lexicographical sorting of $K$ images is more time consuming than the simple linear filtering. For our ordering algorithm the nonlinear filtering requires more time than the sorting step. Nevertheless the running time for our filtering for $\theta_2$ is linear in the number of pixels $n$.

All algorithms are implemented in Matlab 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 $N = 256, 512, 1024, 2048$. The images are presented in Fig. 2. The tables give the average computation time of 50 runs of the algorithms.

We present two numerical experiments:
1. Ordering of natural images: The results are reported in Tab. II. Here Fail gives the percentage of image pixels which cannot be faithfully ordered. The fast VA algorithm with a reduced number of 5 iterations clearly outperforms the LM and WA algorithms.

2. Histogram equalization inversion: first the original 8-bit image $f$ with histogram $h_f$ is mapped to an 8-bit image $g$ which histogram resembles a uniform distribution. This requires the first application of an ordering algorithm. Then $g$ is transformed to an 8-bit image $\tilde{f}$ with histogram $h_{\tilde{f}}$ which requires a second time an ordering algorithm. Tab. III shows the PSNR $20 \log_{10}(255M \cdot N/\|f - \tilde{f}\|_2)$, 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 and WA wrt PSNR and FAIL. Moreover, VA-$\theta_2(5)$ is the fastest algorithm.

The quality of our VA algorithms is emphasized by Fig. 3 which shows three difference images of the original image $f$ and the images $\tilde{f}$ obtained by 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 its competitors in particular at edges.

V. CONCLUSIONS AND FUTURE WORK

In this paper we have proposed a fast and meaningful strict ordering algorithm for integer values in natural images. We intend to apply this algorithm as the basis in various image processing applications, one example is the enhancement of color images in [2]. Further we want to extend the error estimates in Theorem 2 and Corollary 1 to other functionals relevant in imaging and to use these estimates to find appropriate regularization parameters.

Acknowledgements. The work of Mila Nikolova was supported in part by the “FMJH Program Gaspard Monge in optimization and operation research”, and by the support to this program from EDF.

REFERENCES


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


Fig. 3. 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 down: LM, WA, VA-θ₂(35), VA-θ₂(5). The variational methods (VA) contain much less errors than those achieved by LM and WA. Moreover there is no visual difference between VA with 35 iterations and its faster variant with only 5 iterations.