HHS Public Access

Author manuscript

IEEE Trans Med Imaging. Author manuscript; available in PMC 2019 January 01.

Published in final edited form as:

IEEE Trans Med Imaging. 2018 January ; 37(1): 162–172. doi:10.1109/TMI.2017.2760104.

A Direct Algorithm for Optimization Problems with the Huber Penalty

Jingyan Xu,

Division of Medical Imaging Physics, Department of Radiology, Johns Hopkins University

Frédéric Noo, and

Department of Radiology and Imaging Sciences, University of Utah

Benjamin M. W. Tsui

Division of Medical Imaging Physics, Department of Radiology, Johns Hopkins University

Abstract

We present a direct (noniterative) algorithm for one dimensional (1-D) quadratic data fitting with neighboring intensity differences penalized by the Huber function. Applications of such an algorithm include 1-D processing of medical signals, such as smoothing of tissue time concentration curves in kinetic data analysis or sinogram preprocessing, and using it as a subproblem solver for 2-D or 3-D image restoration and reconstruction. Dynamic programming (DP) was used to develop the direct algorithm. The problem was reformulated as a sequence of univariate optimization problems, for $k = 1, \cdots, N$, where N is the number of data points. The solution to the univariate problem at index k is parameterized by the solution at $k + 1$, except at $k = N$. Solving the univariate optimization problem at $k = N$ yields the solution to each problem in the sequence using backtracking. Computational issues and memory cost are discussed in detail. Two numerical studies, tissue concentration curve smoothing and sinogram preprocessing for image reconstruction, are used to validate the direct algorithm and illustrate its practical applications. In the example of 1-D curve smoothing, the efficiency of the direct algorithm is compared with four iterative methods: the iterative coordinate descent, Nesterov's accelerated gradient descent algorithm, FISTA, and an off-the-shelf second order method. The first two methods were applied to the primal problem, the others to the dual problem. The comparisons show that the direct algorithm outperforms all other methods by a significant factor, which rapidly grows with the curvature of the Huber function. The second example, sinogram preprocessing, showed that robustness and speed of the direct algorithm are maintained over a wide range of signal variations, and that noise and streaking artifacts could be reduced with almost no increase in computation time. We also outline the application of the proposed 1-D solver for imaging applications.

Index Terms

denoising; smoothing; total variation; Huber penalty; robust estimation; dynamic programming

I. Introduction

Smoothing or denoising is a ubiquitous problem in one-dimensional (1-D) signal processing. This problem also finds applications in 2-D or 3-D as signal smoothing can be used as subproblem solvers for image restoration and reconstruction. Novel results on signal smoothing were recently published [1]–[4]: direct, noniterative solutions based on dynamic programming (DP) were proposed for 1-D signal smoothing with quadratic data fitting and total variation (TV) penalty. These DP-based solutions are effective for 1-D signal smoothing [1], [2]. When used as subproblem solvers, they also enable fast 2-D image processing [4], [5].

The TV penalty encourages piecewise constant signals and may not be suitable for some applications. One example is smoothing of tissue time concentration curves obtained in dynamic contrast-enhanced imaging. These signals, by their kinetic material exchange nature, are not piecewise constant but smooth with sharp transitions. Another example is sinogram smoothing [6], which preprocesses the sinogram before image reconstruction. The object can often be assumed to be piecewise constant, then the sinogram is smooth with sharp transitions at organ boundaries. Thus there is a need to consider alternative regularization functions that encourage such signal properties. We propose extensions of the previous results [1]–[4] for this purpose. Specifically, we consider quadratic data fitting with the Huber function instead of TV to penalize neighboring intensity differences.

The utility of the Huber penalty in imaging applications is extensive. It can serve as a generic edge preserving penalty [7]–[10], or as an approximation to the nondifferentiable TV penalty [11]–[13] to deal with the computation challenge, or as an effective means of reducing staircasing or the patchy artifacts of the TV penalty [14], [15]. These published works motivate the search for a dedicated solver for problems involving the Huber penalty. Our development is a first step in this direction.

The objective function we seek to minimize is:

$$\Theta(\boldsymbol{x}) \triangleq \sum_{k=1}^{N} \frac{w_k}{2}(y_k - x_k)^2 + \sum_{k=1}^{N-1} \beta_k H(x_k - x_{k+1}, \delta_k), \tag{1}$$

where $\boldsymbol{x} \triangleq \{x_1, \cdots, x_N\}$ is the unknown, N is the signal length, and $\{y_1, \cdots, y_N\}$ is the noisy input signal. The weight, $w_k > 0$, in the data fitting term is typically the inverse variance of the data y_k. The penalty term involves the Huber function with parameter $\delta \geq 0$, namely

$$H(s, \delta) = \begin{cases} \frac{s^2}{2\delta} & |s| \leq \delta \\ |s| - \frac{\delta}{2} & |s| > \delta \end{cases} \tag{2}$$

The parameter δ controls the transition between the quadratic region ($|s| < \delta$) and the linear region ($|s| \geq \delta$). As δ tends to zero, $H(\cdot, \delta)$ approaches the absolute value, $|\cdot|$, so that problem

(1) becomes 1-D quadratic fitting with TV penalty, which was discussed in previous publications [1]–[4], except that here we further allow pixel-dependent weights in the data fitting term as well as in the penalty term, denoted as w_k and $\beta_k > 0$, respectively.

Iterative algorithms can minimize (1). Indeed, the performance of the direct TV algorithm [1] was compared with FISTA [16] as applied to the dual problem of 1-D TV smoothing. It is simple to derive that the dual problem to (1), for $\delta_k \neq 0$ as well as $\delta_k = 0$, is a least squares problem with interval constraints. Then not only are the first order methods such as FISTA applicable, but also the second-order methods such as the active-set method [17]. Also, unlike the TV special case, the Huber penalty makes the (primal) objective (1) a *smooth* convex function, thus both the gradient descent and the coordinate descent algorithms can be applied. In fact, it can be shown that the Huber penalty is a special case of the smoothing technique in [18] for dealing with the nonsmooth TV term, so that the accelerated gradient descent (AGD) method [19] can be used to solve (1).

We emphasize that the existing algorithms to solve (1) are all iterative in nature. The number of iterations will depend on the parameters of the problem, *i.e.*, β_k, δ_k, and the required accuracy of a solution. In contrast, we propose in this work a direct, noniterative algorithm to compute the exact solution to (1), and we demonstrate using extensive numerical comparisons the algorithm's high computational efficiency.

The rest of the paper is organized as follows. Sec II describes the solution to problem (1). In Sec III we introduce four methods (two for solving the primal problem, two for the dual) used for efficiency comparison. Sec IV presents numerical results of applying the direct algorithm to 1-D curve smoothing and sinogram smoothing. Sec V discusses potential future extensions of the 1-D algorithm, including pathways to 2-D (3-D) imaging applications. Sec VI concludes the paper.

II. A DP-based direct algorithm

Since the objective function (1) is strong convex, the existence and uniqueness of a global minimum is guaranteed. As mentioned earlier, we adopted a dynamic programming (DP) approach to find the minimum of (1). An overview of this approach is first given for readers who are not familiar with DP. Next, the solution and the associated algorithm are given in detail, including an analysis of the relationship with 1-D quadratic fitting with TV penalty,

A. Overview

DP transforms the problem of minimizing (1) into a sequence of (simpler) minimization problems [20], [21]. First, we introduce some short hand notation for convenience. Let the unary terms in (1) be $E_k(x_k) \triangleq w_k(y_k - x_k)^2/2$, and $H_k(x_k - x_{k+1}) \triangleq H(x_k - x_{k+1}, \delta_k)$, Expanding all terms in (1), we have

$$\begin{aligned}\min_{\boldsymbol{x}}\Theta(\boldsymbol{x}) \equiv \min_{x_1,x_2,\cdots x_N}\{&E_1(x_1)+\beta_1 H_1(x_1-x_2)\\&+E_2(x_2)+\beta_2 H_2(x_2-x_3)+\cdots\\&+E_{N-1}(x_{N-1})+\beta_{N-1}H_{N-1}(x_{N-1}-x_N)\\&+E_N(x_N)\}.\end{aligned} \tag{3}$$

Since x_1 only appears in the first two terms of (3), its influence on the solution can be summarized by defining a new function, Φ_2, where

$$\Phi_2(x_2) \triangleq \min_{x_1}\{E_1(x_1)+\beta_1 H_1(x_1-x_2)\}. \tag{4}$$

Due to the interaction between x_1 and x_2 from the Huber function, the solution to (4), x_1^*, depends on x_2 and so does the function value Φ_2. We can use (4) to eliminate all occurrences of x_1 in (3), which becomes

$$\min_{\boldsymbol{x}}\Theta(\boldsymbol{x}) \equiv \min_{x_2,\cdots x_N}\{\Phi_2(x_2)+E_2(x_2)+\beta_2 H_2(x_2-x_3)+\cdots+E_{N-1}(x_{N-1})+\beta_{N-1}H_{N-1}(x_{N-1}-x_N)+E_N(x_N)\}, \tag{5}$$

Doing so, we have reduced the original N-variable minimization problem of equation (3) to a minimization problem with $N-1$ variables. This procedure can be repeated by introducing

$$\Phi_{k+1}(x_{k+1})=\min_{x_k}\{\Phi_k(x_k)+E_k(x_k)+\beta_k H_k(x_k-x_{k+1})\} \tag{6}$$

for $k = 1, \cdots, N-1$, each time reducing the problem size by one. For consistency, we let $\Phi_1(x_1) \equiv 0$ in (4). At the last step, we have the following univariate minimization problem

$$\min_{\boldsymbol{x}}\Theta(\boldsymbol{x}) \equiv \min_{x_N}\{\Phi_N(x_N)+E_N(x_N)\}. \tag{7}$$

Thus the solution to (3) is equivalent to solving (a) the sequence of univariate minimization problems in (6), whose solution depends on the "future" variable x_{k+1}, and (b) the terminal problem, (7). Both (a) and (b) admit closed-form solutions because the minimization problems only involve piecewise quadratic convex functions. This claim can be established by induction after studying the solution to (6).

B. Solution to (6)

To reduce notation clutter, we first rewrite (6) as the following generic problem, which omits dependence on k:

$$\Psi(t)=\min_s \Omega(s,t) \quad (8a)$$

$$\text{where } \Omega(s,t) \triangleq \underline{\Phi(s)+E(s)}+\beta H(s-t) \\ =\underline{Q(s)}+\beta H(s-t) \quad (8b)$$

The unary function $E(s) \triangleq w(y-s)^2/2$ is quadratic. We also define $Q(s) \triangleq \Phi(s) + E(s)$, and implicitly $H(\cdot) \equiv H(\cdot, \delta)$.

It is easy to see that if $\Phi(s)$ is convex, then $\Omega(s, t)$ is (strongly) convex in (s, t). From convex analysis, $\Psi(t)$ in (8a) is convex in t; see *e.g.* [22, page 51] or [23, page 87]. Also, there exists a unique minimizer s^* such that

$$s^*=\arg\min_s \{Q(s)+\beta H(s-t)\} \quad (9)$$

Note that s^* is a function of t, but this dependence is not written explicitly.

Assume for now that Φ is continuously differentiable, convex, and piecewise quadratic with a finite number, B, of knots. Its gradient, ϕ, is then piecewise linear, continuous, and non-decreasing, which implies that ϕ is fully characterized by its knots: $(s_i^b, \phi(s_i^b))$, $i = 1, \cdots, B$. That is,

$$\phi(s)=\frac{s_{i+1}^b-s}{s_{i+1}^b-s_i^b}\phi(s_i^b)+\frac{s-s_i^b}{s_{i+1}^b-s_i^b}\phi(s_{i+1}^b) \quad s_i^b \le s<s_{i+1}^b$$

Let the gradient of $Q(s)$ be $q(s)$, then $q(s)$ is also piecewise linear and continuous with knots at locations s_i^b and $q(s_i^b)=\phi(s_i^b)+w(s_i^b-y)$, $i = 1 \cdots B$. Moreover, $q(s)$ is increasing and unbounded, *i.e.*, $q(s) \to -\infty$ as $s \to -\infty$, and $q(s) \to \infty$ as $s \to \infty$. Similar to $\phi(s)$, the value of $q(s)$ at any s can be obtained using a linear interpolation formula.

The solution (9) satisfies

$$0=q(s^*)+\beta h(s^*-t,\delta) \quad (10)$$

where $h(\cdot)$ is the gradient of the Huber function (2), i.e.,

$$h(s)=\begin{cases} 1 & s>\delta \\ s/\delta & |s| \le \delta \\ -1 & s<-\delta \end{cases}$$

Note that, as with $H(\cdot)$, the dependence of h on δ is not written explicitly in this section. Since h is bounded between ±1, $q(s^*) \in [-\beta, \beta]$ in (10). As $q(s)$ is piecewise linear, strictly increasing, and unbounded, there exists s^- and s^+, and $s^- < s^+$, such that

$$q(s^+)=\beta, \quad (11a)$$

$$q(s^-)=-\beta \quad (11b)$$

and $s^* \in [s^-, s^+]$. Whereas s^* depends on t, s^- and s^+ are independent of t since they are determined by $q(s)$ only.

A closed-form expression for s^* can be obtained by considering t in three cases as illustrated in Fig. 1.

1. If $t - \delta \ge s^+$, then $s^* = s^+$. As s^* remains a constant, and $s^+ - t \le -\delta$, $\Psi(t) = Q(s^+) + \beta H(s^+ - t)$ is a linear function of t with gradient equal to β.
2. If $t + \delta \le s^-$, $s^* = s^-$ and $\Psi(t) = Q(s^-) + \beta H(s^- - t)$ is linear in t, with gradient equal to $-\beta$.
3. If $s^- - \delta < t < \delta + s^+$, then $s^* \in (t - \delta, t + \delta)$, and $q(s^*) \in (-\beta, \beta)$. Thus, the Huber function operates in the quadratic region, and s^* satisfies [Fig. 1(c)]

$$q(s^*)+\beta(s^*-t)/\delta=0, \quad (12)$$

which is equivalent to

$$t=s^*+\frac{\delta}{\beta}q(s^*), \quad (13a)$$

$$s^*=\left[1+\frac{\delta}{\beta}q\right]^{-1} t \quad (13b)$$

Given the properties of q, s^* is a piecewise linear, continuous and increasing function of t, and the knots of s^* are at positions

$$t_i^b = s_i^b + \frac{\delta}{\beta} q(s_i^b) \quad \forall i \text{ s. t. } s^- < s_i^b < s^+. \tag{14}$$

Now we evaluate

$$\Psi(t) = Q(s^*) + \beta(s^* - t)^2/(2\delta), \tag{15}$$

using its gradient $\psi(t)$. By the chain rule,

$$\psi(t) = \frac{\partial \Psi}{\partial s^*}\frac{ds^*}{dt} + \frac{\partial \Psi}{\partial t}. \tag{16}$$

Since $\partial\Psi/\partial s^* = 0$ from (12), we have

$$\psi(t) = -\frac{\beta}{\delta}(s^* - t) \tag{17a}$$

$$\overset{(12)}{=} q(s^*). \tag{17b}$$

Thus $\psi(t)$ is a piecewise linear, continuous and increasing function of t with knots at t_i^b (14). As t increases from $s^- - \delta$ to $s^+ + \delta$, $\psi(t)$ increases from $-\beta$ to β. Combining (17a) with (14), we get the gradient value at the knots:

$$\psi(t_i^b) = -\frac{\beta}{\delta}(s_i^b - t_i^b) = q(s_i^b). \tag{18}$$

(18) and (14) characterize the piecewise linear $\psi(t)$.

Summarizing key points from all three cases, we have $s^* = s^-$ when $t \le s^- - \delta$ and $s^* = s^+$ when $t \ge s^+ + \delta$; otherwise, s^* is a piecewise linear function of t. Correspondingly, $\Psi(t)$ is linear for $t \le s^- - \delta$ and $t \ge s^+ + \delta$, and piecewise quadratic otherwise. Also, $\Psi(t)$ is continuously differentiable and convex. Its gradient $\psi(t)$ is bounded below by $-\beta$ and above by β; otherwise it is piecewise linear and continuous. Combine (13) and (17), we have

$$\psi(t)|_{t = s + \frac{\delta}{\beta} q(s)} = q(s), \quad s^- < s < s^+ \tag{19}$$

Thus $\psi(t)$ is obtained by clipping $q(s)$ at $\pm\beta$, and relabeling the axis (and the knots) according to (19), a stretching operation anchored at the point s_0 where $q(s_0) = 0$. These key

points are depicted in Fig. 2. Further, If $\phi(s)$ has B knots, then $\psi(t)$ has at most $B+2$ knots, two of which are at $(s^- - \delta, -\beta)$ and $(s^+ + \delta, \beta)$, whereas the others, $(t_i^b, \psi(t_i^b))$, are transferred from the knots $(s_i^b, q(s_i^b))$ by (14) and (18).

C. Algorithm description

The analysis of Sec II-B can be repeatedly applied to (6), by making the substitutions for each $k = 1, \cdots N-1$.

$$\begin{aligned} s{:=}x_k, \quad s^*{:=}x_k^*, \quad & t{:=}x_{k+1}, \\ \Phi(s){:=}\Phi_k(x_k), \quad & \Psi(t){:=}\Phi_{k+1}(x_{k+1}), \\ \phi(s){:=}\phi_k(x_k), \quad & \psi(t){:=}\phi_{k+1}(x_{k+1}), \end{aligned}$$

starting from $\Phi_1(\cdot) \equiv 0$ which is continuously differentiable, convex, and piecewise quadratic with a finite number of knots. We also define $x_k^{\pm}$ and $q_k(x_k)$ such that

$$\begin{aligned} q_k(x_k^+) = \beta_k, \quad q_k(x_k^-) = -\beta_k, \\ q_k(x_k) = \phi_k(x_k) + w_k(x_k - y_k). \end{aligned}$$

Thus, for all $k = 1, \cdots, N$, Φ_k is a continuous, convex, and piecewise quadratic function, whose gradient is bounded between $[-\beta_k, \beta_k]$ and piecewise linear with knots $(x_k^{b,i}, \phi_k(x_k^{b,i}))$.

The DP-based algorithm consists of two sweeps, a forward sweep that builds the list of knots $(x_k^{b,i}, \phi_k(x_k^{b,i}))$, $k = 1, \cdots N$, as the sequence continues, and a backward sweep to trace back values of x_k^*, $k < N$ once x_N^* is obtained at the end of the forward sweep. Now we discuss in more detail the two sweeps.

Assume we know $(x_k^{b,i}, \phi_k(x_k^{b,i}))$, which are the knots for $\phi_k(x_k)$. We obtain $(x_{k+1}^{b,i}, \phi_{k+1}(x_{k+1}^{b,i}))$ by applying two steps to the knots of $\phi_k(x_k)$, (1) trimming and (2) transfer, and by inserting two knots coming from $q_k(s) = \pm\beta_k$. In the trimming step, we identify and retain the knots of $\phi_k(x_k)$ for which $s = x_k^{b,i}$ yields $q_k(s) = \phi_k(s) + w_k(s - y_k) \in (-\beta_k, \beta_k)$, see Fig. 3. In the transfer step, we use (14) and (18) to transfer the retained knots $x_k^{b,i}$ to $x_{k+1}^{b,i}$, and $x_k^{\pm}$ to $x_k^{\pm} \pm \delta_k$, and also copy the knot's function value $q_k(x_k^{b,i})$ to $\phi_{k+1}(x_{k+1}^{b,i})$.

At $k = N$, we need the solution to (7), *i.e.*, x_N^* such that $q_N(x_N^*) = 0 = \phi_N(x_N^*) + w_N(x_N^* - y_N)$. Since $q_N(\cdot)$ is piecewise linear, continuous and increasing, we can obtain the exact x_N^* by searching for the zero-crossing linear section of $q_N(x_N)$ and linear interpolation. We will also need $\phi_N(x_N^*)$ to initiate the backtracking process, which is obtained as $\phi_N(x_N^*) = -w_N(x_N^* - y_N)$.

Once x_N^* is obtained, we trace back values x_k^*, $k = N-1, \cdots 1$ following the three cases in Sec II-B.

1. If $x_{k+1}^* - \delta_k \geq x_k^+$, then $x_k^* = x_k^+$, and since $\beta_k = q_k(x_k^+) = \phi_k(x_k^+) + w_k(x_k^+ - y_k)$ we have

$$\phi_k(x_k^*) = \beta_k - w_k(x_k^* - y_k). \quad (20)$$

2. If $x_{k+1}^* + \delta_k \leq x_k^-$, then $x_k^* = x_k^-$, and since $-\beta_k = q_k(x_k^-) = \phi_k(x_k^-) + w_k(x_k^- - y_k)$, we have

$$\phi_k(x_k^*) = -\beta_k - w_k(x_k^* - y_k). \quad (21)$$

3. If $x_k^- - \delta_k < x_{k+1}^* < \delta_k + x_k^+$, then from (17)

$$x_k^* \overset{(17a)}{=} x_{k+1}^* - \frac{\delta_k}{\beta_k}\phi_{k+1}(x_{k+1}^*) \quad (22)$$

$$\begin{aligned} \phi_k(x_k^*) &= q_k(x_k^*) - w_k(x_k^* - y_k) \\ &\overset{(17b)}{=} \phi_{k+1}(x_{k+1}^*) - w_k(x_k^* - y_k) \end{aligned} \quad (23)$$

To summarize, the backtracking of x_k^* is

$$x_k^* = \begin{cases} x_k^-, & x_{k+1}^* < x_k^- - \delta_k \\ F_{k+1}(x_{k+1}^*) & x_k^- - \delta_k \leq x_{k+1}^* \leq x_k^+ + \delta_k \\ x_k^+, & x_k^+ + \delta_k < x_{k+1}^* \end{cases} \quad (24)$$

where $F_{k+1}(u) = u - \frac{\delta_k}{\beta_k}\phi_{k+1}(u)$. We recursively apply (24) until $k = 1$. At the same time, we also need to calculate $\phi_k(x_k^*)$.

D. A special case: TV smoothing

When $\delta_k = 0$, (1) reduces to 1-D smoothing with quadratic data fitting and a TV penalty. This special case simplifies the analysis and algorithm implementation as there is a simpler relation for the knots transfer. Considering $\delta_k = 0$ and Fig. 1, we need to make some adjustment to the analysis in Sec II-B as follows.

1. If $x_{k+1} \geq x_k^+$, then $x_k^* = x_k^+$,

$$\Phi_{k+1}(x) = Q(x_k^+) + \beta_k(x_{k+1} - x_k^+)$$

which is linear with slope β_k.

2. If $x_{k+1} \leq x_k^-$, then $x_k^* = x_k^-$, and

$$\Phi_{k+1}(x) = Q(x_k^-) + \beta_k(x_k^- - x_{k+1})$$

which is linear with slope $-\beta_k$.

3. If $x_k^- < x_{k+1} < x_k^+$, then $x_k^* = x_{k+1}$ as this value gives $q_k(x_k^*) \in (-\beta_k, \beta_k)$ and $q_k(x_k^*) + \beta_k \partial |x_k^* - x_{k+1}| = 0$, where $\partial|\cdot - x_{k+1}|$ is the subgradient of the absolute value function evaluated at x_k^*. As $x_k^* = x_{k+1}$, we have

$$\Phi_{k+1}(x_{k+1}) = \Phi_k(x_{k+1}) + E_k(x_{k+1}), \quad (25)$$

From (25), the knot locations of $\Phi_{k+1}(x_{k+1})$ are exactly those of $\Phi_k(x_k)$, *i.e.*, $x_{k+1}^{b,i} = x_k^{b,i}$.

Summarizing all three cases, the solution x_k^* is again a piecewise linear function of x_{k+1}, but it contains *exactly* 3 linear sections: (1) $x_k^* = x_k^-$, (2) $x_k^* = x_{k+1}$, and (3) $x_k^* = x_k^+$. Using Fig. 2(a) as a reference, the complicated multi-section middle portion is compressed to overlap with the diagonal line $s = t$ as $\delta \to 0$. This observation has implications for the difference in the worst case memory cost and computational complexity between the Huber penalty and the TV penalty.

During the forward sweep, we build the list of knots by

$$x_{k+1}^{b,i} = x_k^{b,i}, \quad \text{where } x_k^- < x_k^{b,i} < x_k^+ \quad (26)$$

$$\phi_{k+1}(x_{k+1}^{b,i}) = \phi_k(x_k^{b,i}) + w_k(x_k^{b,i} - y_k) = q_k(x_k^{b,i}) \quad (27)$$

and by inserting the two incoming knots at x_k^- and x_k^+. Note that (26) can be obtained from (14) by setting $\delta_k = 0$, and (27) is the same as (18).

To obtain the solution, we need all knots of $\phi_N(x_N)$ to search for x_N^* such that $q_N(x_N^*) = \phi_N(x_N^*) + w_N(x_N^* - y_N) = 0$. Afterwards, tracing back values x_k^*, for $k = N - 1, \cdots, 1$ is much simpler as we have

$$x_k^* = \begin{cases} x_k^-, & x_{k+1}^* < x_k^- \\ x_{k+1}^*, & x_k^- < x_{k+1}^* \le x_k^+ \\ x_k^+, & x_k^+ < x_{k+1}^* \end{cases} \quad (28)$$

This special case discussion is in essence Algorithm-1 in [2], except that here we consider weighted data fitting terms and possibly pixel-dependent TV parameter β_k.

Comparing (28) and (24), we notice that by setting $\delta_k = 0$ in (24), the DP algorithm for the Huber penalty immediately reduces to the DP algorithm for TV penalty. This was used to generate the numerical results that are based on TV in Sec IV.

E. Implementation, memory and computational cost

One way to implement (24) is to combine it with (20),(21) and (23). To do this, we need to keep track of (a) the set of knots ($x_k^{b,i}, \phi_k(x_k^{b,i})$) at $k = N$ for searching x_N^*, and (b) the boundary values, x_k^- and x_k^+, for $k = 1, \cdots N-1$. In the worst case, the number of knots grows linearly in k. Thus the worst case memory requirement for implementing (24) is $O(N)$, the same as for the TV penalty [2], [4].

However, we have observed that the implementation just described is not always numerically stable, especially for large values of δ_k. A possible reason can be inferred from the recursion (23) as follows. Combine (22) with (23) to eliminate x_k^*, then (23) is equivalent to

$$\phi_k(x_k^*) = (1 + \delta_k \frac{w_k}{\beta_k}) \phi_{k+1}(x_{k+1}^*) - w_k(x_{k+1}^* - y_k). \quad (29)$$

In (29), the multiplicative factor of $\phi_{k+1}(x_{k+1}^*)$ is larger than 1 for $\delta_k > 0$, thus any round-off error of $\phi_{k+1}(x_{k+1}^*)$ will grow as k decreases, which appears consistent with our observations.

To avoid the numerical stability issue, we did not use (23) to back-track $\phi_k(x_k^*)$. Instead, after obtaining x_k^* using (22), we use linear interpolation to calculate $\phi_k(x_k^*)$ from the knots locations ($x_k^{b,i}, \phi_k^{b,i}(x_k^{b,i})$). This method was numerically validated to be stable and was used to produce all results in this paper. But this numerical stability comes at a memory cost. Instead of the worst case $O(N)$ memory cost, we now have a worst case $O(N^2)$ memory cost, corresponding to keeping all knots in memory: ($x_k^{b,i}, \phi_k^{b,i}(x_k^{b,i})$) for all $k = 1, \cdots N$. For TV smoothing, since $\delta_k = 0$, and since the recursion (28) does not need to keep track of $\phi_k(x_k^*)$, this issue of numerical instability does not exist.

In practice, as we only need to consider the range $x_k \in [x_k^-, x_k^+]$ for which $q_k(x_k) \in [-\beta_k, \beta_k]$, the number of knots does not necessarily increase with the sequence length (the trimming step, Fig 3). Thus, the empirical memory cost will be usually much smaller than

$O(N^2)$. For instance, the maximum number of knots (for all k) in the example of Fig. 4 was 4 even though $N = 400$.

Now we discuss computational cost. Defining an operation as either a multiplication, an addition, or a comparison, we estimate the computational cost to be $O(\Sigma_k B_k)$, where B_k is the number of knots at each index $k = 1, \ldots, N$. This is independent of the method used to back-track $\phi_k(x_k^*)$. In the worst case when B_k grows linearly, the cost can thus be $O(N^2)$. However, just as with the memory cost discussion, in practice, B_k may often remain small and essentially independent of N, as observed in our numerical studies, in which case the effort becomes $O(N)$.

III. Methods for comparison

In Sec IV-A, our direct algorithm is compared with four iterative methods for efficiency evaluation. Two methods were used to solve the primal problem, namely iterative coordinate descent (ICD) and Nesterov's accelerated gradient descent (AGD) [19], and two for the dual problem. When $\delta_k > 0$, $\forall k$, the objective function (1) is convex and smooth, thus both ICD and accelerated gradient algorithms converge to the global minimum [24], [25]. To apply Nesterov's method, the Lipschitz constant of the gradient of the objective (1) was estimated as $L_p = \|\{\boldsymbol{w}\}_d + D^T\{\boldsymbol{\beta}/\boldsymbol{\delta}\}_d D\|_2$, where

$$\{\boldsymbol{w}\}_d = \mathrm{diag}\{w_1, \cdots, w_N\} \quad (30)$$

$$\{\boldsymbol{\beta}/\boldsymbol{\delta}\}_d = \mathrm{diag}\{\beta_1/\delta_1, \cdots, \beta_{N-1}/\delta_{N-1}\} \quad (31)$$

and D is the finite difference operator, *i.e.*,

$$D\boldsymbol{x} = [x_1 - x_2, \cdots, x_{N-1} - x_N]^T \quad (32)$$

for $\boldsymbol{x} = \{x_1, \cdots, x_N\}$. We point out that the smaller the smoothing constant δ (if $\delta_k = \delta > 0$), the larger the Lipschitz constant L_p, which reduces the step size and also the convergence speed.

To obtain the dual problem to (1), we start from the following equivalent expression for the Huber function (2),

$$H(s,\delta) = \inf_t \left\{ |t| + \frac{(s-t)^2}{2\delta} \right\} = \max_{|t|\le 1} \{st - \frac{\delta}{2}t^2\}. \quad (33)$$

The first equality in (33) can be obtained by direct evaluation from the minimizer (a soft-thresholding on s), see, *e.g.*, [26]; the second equality in (33) can be derived using the conjugate function of the Moreau envelope (the Moreau-Yosida regularization) [27, page 135] of the function $|\cdot|$.

We plug (33) into the objective function (1), switch the order of minimization and maximization, and upon minimizing with respect to x, we obtain the following dual minimization problem:[1]

$$\min \{\Psi(\boldsymbol{u}) = \frac{1}{2}\boldsymbol{u}^T \left(\{\boldsymbol{\beta}/\boldsymbol{\delta}\}_d^{-1} + D\{\boldsymbol{w}\}_d^{-1} D^T\right) \boldsymbol{u} - \boldsymbol{u}^T D\boldsymbol{y}\} \quad \text{subject to} \quad |u_k| \leq \beta_k, \quad \forall k = 1, \cdots, N-1. \tag{34}$$

and $\boldsymbol{u} = \{u_1, \cdots, u_{N-1}\}$, $\boldsymbol{y} = \{y_1, \cdots, y_N\}$. The primal optimal x^* and the dual optimal u^* are related by

$$\boldsymbol{x}^* = \boldsymbol{y} - \{w\}_d^{-1} D^T \boldsymbol{u}^* \tag{35}$$

The problem (34) is a least squares problem with simple interval constraints. Thus, second order algorithms such as those in quadratic programming are applicable. For the numerical examples in IV-A, we applied Matlab implementation (R2015b) of `lsqlin`, a built-in function for constrained least squares minimization. We compared the three available algorithm options `trust region`, `interior point` and `active set` and found that the `active set` option (with analytical gradient and Hessian) consistently performed the best. This option executes the quadratic programming active-set algorithm [17], [28] that at each iteration solves an unconstrained least squares problem. In addition to applying this off-the-shelf algorithm, we also implemented FISTA [16] to solve (34). The required Lipschitz constant was estimated as $L_d = \|\{\boldsymbol{\beta}/\boldsymbol{\delta}\}_d^{-1} + D\{w\}_d^{-1} D^T\|_2$.

To initialize the two primal algorithms (ICD and Nesterov's AGD), we set $\boldsymbol{x}_0 = \boldsymbol{y}$, *i.e.*, the initial value was the same as the input; to initialize the dual algorithms, we set $\boldsymbol{u}_0 = \boldsymbol{0}$, a vector of all zeros which is a feasible solution to (34). From (35), these two settings are compatible.

IV. Numerical results

We consider two example applications of the direct 1-D algorithm, tissue concentration curve smoothing in dynamic contrast-enhanced MR studies and sinogram smoothing, which

[1] The dual variable u is related to t in (33) by $u_k = \beta_k t_k$.

are both effective preprocessing steps to improve subsequent kinetic data analysis or image reconstruction.

A. Smoothing of 1-D signal

The noisy tissue curves were generated using a parametric model for brain kinetic data analysis [29]. The signal length was $N = 400$. The curves were denoised using $w_k = 1$, $\beta_k = 0.1$, and $\delta_k = \delta$ for all k. We experimented with four δ values, 0.1, 10^{-2}, 10^{-3}, and 10^{-4}. For performance evaluation, we generated 10 noise realizations for each δ.

Fig. 4 (a) shows a sample denoising result from one noise realization when $\delta = 0.01$. Comparing with the TV smoothing result in Fig 4(b), *i.e.*, $\delta = 0$, the characteristic staircase artifact in Fig. 4(b) is absent in Fig. 4(a), which is smooth and physiologically more reasonable.

We use the direct DP solution to assess convergence speed of the four methods in Sec III. For the three in-house methods, ICD, Nesterov's AGD, FISTA applied to the dual problem (FISTA-d), we used the maximum absolute difference (MAD) over all data points ε between the iterative solution and the direct DP solution as a stopping criterion, and recorded the number of iterations and computation time when ε reached below 10^{-10}. For Matlab implementation of the `lsqlin` algorithm with `active set` option, we set maximum iteration number to be 100, and other tolerance settings to default values. We then calculated the MAD between the output of the algorithm and the direct DP solution and report the computation time and iteration numbers.

Fig. 4(c) and (d) show the MAD versus iteration numbers (c) and wall clock time (d) for each combination of δ and the iterative method, with clustered symbols representing the 10 noise realizations. The wall clock time results take into account the complexity of the four algorithms, which is critical here since second order methods are computationally more costly per iteration than first order methods.

Fig. 4(c) shows that the second order active set method requires far fewer iterations than the other three methods to reach an accuracy of 10^{-15}. For $\delta = 0.1$, there are problem instances such that the active set method converged in one iteration. Moreover, the convergence of all four methods is affected by δ: the smaller the value of δ, the more iterations it takes to converge.

As shown in Fig. 4(d), the advantage of the active set method is much diminished once the computational complexity is taken into account, as one iteration of the active set method (0.1 sec) is already as costly as hundreds of iterations of FISTA. For reference, the vertical dashed line in Fig. 4(d) marks the runtime for the DP algorithm (0.012 sec), which is more than 8 times faster than one iteration of active set. For $\delta = 0.01$, which was used to create the result in (a), the fastest method out of the four turns out to be FISTA-d. Depending on the noise realization, this method is still 9–15 times slower than the DP algorithm. It is also obvious from Fig. 4(c) and (d) that ICD is competitive with FISTA or AGD if measured by number of iterations, but its sequential update nature makes it unfavorable in terms of runtime.[2]

As a measure of the memory cost of the DP algorithm, we looked at how the number of knots B_k varied with the sequence index k. In general, the number of knots depends on the parameter values, *i.e.*, δ and β. For the Huber smoothing result in Fig. 4(a), the progression of B_k as a function of the sequence index k is plotted in Fig. 5. Due to the trimming and knots removal, the number of knots does not necessarily increase as the sequence progresses. The maximum number of knots in this example was 4.

B. Sinogram smoothing

The second example was sinogram smoothing followed by image reconstruction. The fan-beam sinograms of a physical phantom obtained for a previous study [30], [31] were reused here. The data were acquired on a Siemens SOMATOM® Sensation™ 64 CT scanner with the x-ray tube settings of 25 mAs and 120 kVp. The acquisition was repeated to obtain 186 noise realizations. More details of the acquisitions and of the phantom can be found in our previous publications [30], [31].

The phantom data were reconstructed using a fanbeam filtered-backprojection (FFBP) algorithm with and without sinogram smoothing. Internally, the FFBP algorithm performed fan-to-parallel beam rebinning, filtering of the parallel beam data, and backprojection in the parallel geometry. We applied sinogram smoothing to the ramp-filtered parallel beam data (no apodization) before backprojection.

Note that fan-to-parallel beam rebinning and ramp filtering induce slight correlations. Here, we assume these correlations can be neglected and apply objective function (1) to such preprocessed data, acknowledging that such an approach could potentially be sub-optimal.

Conventionally, sinogram smoothing is applied to the original data rather than the filtered data. Our approach is justified for objects that can be reasonably modeled as a combination of pill-box like elemental objects. Indeed, the filtered parallel-beam projection of such an elemental object is constant over the object support and slowly-varying outside, with sharp transitions between the two regions. Such signal features are well amenable to using the Huber penalty as in (1).

To estimate the statistical weights w_k in (1), the empirical variance of the 186 filtered sinograms was calculated. The inverse variance was smoothed by a Gaussian function, then the result was normalized to have a mean value of one and used as the statistical weights.

An example of FFBP reconstruction obtained after sinogram preprocessing is shown side-by-side with the original FFBP image in Fig. 6(a). There is a substantial reduction in noise magnitude and in the horizontal noise streaks connecting the arms.

To quantitatively compare image quality, we calculated task performance versus resolution tradeoffs using (1) different parameter settings for the Huber penalty, and (2) different apodization windows in the FFBP algorithm. The task was estimation of the mean intensity

[2]AGD and FISTA relied on Matlab's parallel implementation of matrix-vector or vector-vector dot product. In a sequential implementation of such operations (as is often done in C), the wall clock time comparison between AGD, FISTA and ICD should be similar to the comparison in iterations.

of a 7 × 7 region-of-interest (ROI). The task performance metric was the ensemble RMSE for this mean intensity estimated from the 186 noise realizations. The average of RMSE results obtained for the 16 ROIs shown as yellow boxes in Fig. 6(a) was used as final figure of merit. The ground truth (needed for the bias component of RMSE) was estimated by applying the original FFBP reconstruction with no apodization to the average of the 186 noise realizations. For image resolution, we extracted and averaged radial profiles across the 200 HU insert (Fig. 6(a), green box) to obtain the modulation transfer function (MTF). The frequency where the MTF is at 50% of its maximum was used as a metric for resolution.

Fig. 6(b) compares the tradeoff curves generated by varying β for different values of δ, from $\delta = 0$ to $\delta = 1$.[3] The comparison includes the tradeoff curve for variations in the FFBP apodization window with no sinogram preprocessing. At high resolution (region A), we see that the Huber penalty with large δ provides higher task performance (lower RMSE). At moderate resolution (region B), using a smaller edge-preserving δ, around 0.02, is more favorable. Additional data (not shown) demonstrate that the tradeoff curve changes continuously as δ varies. For the parameters used for the image in Fig 6(a), $\beta = 0.005$ and $\delta = 0.02$, the gain in RMSE over FFBP is about 15%.

The original sinogram (before smoothing) contains a wide range of signal variations. Fig. 7 plots the maximum number of knots $\max_k B_k$ calculated for each projection view; in the worse case, this number was 7. Again, the number of knots depends on the β and δ values. The small number of knots makes sinogram smoothing using the direct algorithm very efficient. The total time for processing all 1160 views was 600 msec, thus negligible compared to the reconstruction time of analytic FFBP (17 sec), using a single CPU implementation (Intel Xeon® CPU, 2.4 GHz, 48 GB RAM).[4]

V. Discussion

The worst case computational complexity of our algorithm is $O(N^2)$. In the literature, the direct TV algorithm of Condat [1] has a worst case complexity of $O(N^2)$. But the direct algorithms presented in Johnson [2] and Kolmogorov et. al. [4] both have a worst case complexity of $O(N)$; these algorithms are based on a reparameterization of the knots. Direct adoption of such reparameterization will not reduce the computational complexity for the Huber penalty, because the location of the knots will change, and relabeling all knots at each index k, with the worst case linear increase of knots, will still incur an $O(N^2)$ computation. Condat [1] gave a numerical example with the worst case complexity for his algorithm. We empirically tested that, when δ_k's are small, signals with similar characteristics (a linear signal with a slope of $O(1/N^2)$) will have $O(N)$ knots, hence $O(N^2)$ complexity. Such signals are obviously contrived. In our numerical studies, the number of knots was not a function of the signal length, and the numerical complexity was $O(N)$.

[3]When $\delta = 1$, we essentially have quadratic smoothing as very few neighboring differences of the filtered line integrals are outside the range [−1, 1]. Further increase of δ did not change the tradeoff curve.

[4]The speed difference between this and those in Fig. 4 is due to the computation platforms. The DP implementation for producing Fig. 4 was based on Matlab. The sinogram smoothing example was implemented in C.

The several existing DP-based algorithms for TV penalized quadratic fitting can be divided into the primal approach, as in this work, or the dual approach [1], [3]. A dual approach using the objective function (34) can be developed similar to [3]. An interesting topic for future research could be to compare the numerical properties of both primal and dual algorithms.

There are a number of interesting aspects to our development. First, our direct algorithm calculates the exact solution, unlike any other existing ones which are iterative in nature. Researchers can use the output of the direct algorithm as a benchmark for iterative algorithms. Our comparisons demonstrate strong computational advantages over iterative counterparts. Note that these advantages will remain if an alternative penalty is used, such as $\sqrt{\varepsilon^2+s^2}$, which is twice-differentiable and thus enables the use of second-order methods. We have conducted a number of experiments (not reported here) that support this claim.

Notably, the Huber function is the Moreau envelope of the absolute value function. The Huber penalty indeed comes out as a special case of applying the smoothing technique of Nesterov [18, eqn. (4.17)] to the TV penalty. Such a connection between Huber and TV penalties does not exist for other mollified versions of TV, *e.g.*, $\sqrt{\varepsilon^2+s^2}$.

The examples we presented, namely tissue concentration curve smoothing and sinogram smoothing, are two promising applications of the proposed algorithm. For each application, more systematic and task-performance based parameter tuning is warranted to exploit the benefit in image (signal) quality. Our focus in this paper was algorithm development, not evaluation. Regarding sinogram smoothing, we do not currently have evidence showing that denoising the filtered data rather than the original data provides better results. It would be interesting to compare the performance of our approach to that in [6], where the image quality benefits of sinogram smoothing with Huber penalty were thoroughly demonstrated.

Another important application for our algorithm is as a subproblem solver in an iterative algorithm for 2-D or 3-D applications such as image restoration or reconstruction. Such an approach has been suggested for 2-D denoising using the TV penalty [4], [5]; a substantial speed advantage was achieved over the primal-dual algorithm [4], [32].

Here we outline the idea using x-ray CT image reconstruction. The objective function can be written as

$$\frac{1}{2}\|\boldsymbol{y}-A\boldsymbol{x}\|^2+\beta\sum_{i,j\in N_i}H(x_i-x_j). \tag{36}$$

where $\boldsymbol{x}=\{x_i\}$, x_i is the unknown linear attenuation coefficient at pixel i, $i=1,\cdots,N_xN_y$ is the linear index of the pixel, and N_x and N_y are the number of rows and columns of a 2-D image. We use 2-D reconstruction as an example for its notational convenience; extension to 3-D problems is straightforward. The input $\boldsymbol{y}$ is the set of line integrals, and A is the forward projection operator. The Huber penalty H penalizes intensity differences within a pixel's

neighborhood N_i. For 2D problems with a first-order neighborhood, the Huber penalty can be expanded as

$$\sum_{i,j\in N_i} H(x_i - x_j) = \sum_{i',j'} [H(x_{i'+1,j'} - x_{i',j'}) + H(x_{i',j'+1} - x_{i',j'})] \quad (37)$$

where we switched from the linear indexing to the row, column indexing i', j', $i' = 1, \cdots, N_x$, $j' = 1, \cdots, N_y$. To apply our direct algorithm as a subproblem solver, we introduce auxiliary variables x^1, x^2 together with the following constraints

$$x = x^1,\ x = x^2. \quad (38)$$

Then minimizing (36) is equivalent to

$$\min_{x,x^1,x^2} \frac{1}{2}\|y - Ax\|^2 + \beta \sum_{i',j'} [H(x^1_{i'+1,j'} - x^1_{i',j'}) + H(x^2_{i',j'+1} - x^2_{i',j'})], \quad (39)$$

subject to (38).

Note that the Huber penalty in the row (i') and column (j') directions is applied to each individual auxiliary variable x^1, x^2, respectively. If we apply the alternating direction method of multipliers (ADMM) to the above constrained problem, then the subproblems involve 1-D Huber smoothing that are column-wise (j') or row-wise (i') separable, to which our proposed algorithm can be directly applied.

We implemented this idea for a special case of (36) where A is the identity matrix, which means we considered an image denoising problem. We ran the algorithm with 1000 iterations to define the converged solution. Then, we calculated the RMSE for the first 100 iterations. For comparison, we also ran the accelerated gradient descent (AGD) algorithm on the same image. The denoised image obtained with $\delta = 5$ HU and $\beta = 6.7$ is shown together with the original image in Fig. 8. The convergence plot is shown in Fig. 9: it respectively takes 6 and 44 iterations for the proposed method and AGD to reach an accuracy of 0.01 HU. For higher accuracy requirements, the difference is even bigger.

Further extensions of the DP-based algorithms could consider generalized unary terms, such as piecewise linear or piecewise quadratic functions. For example, the quadratic unary terms could be replaced by the Huber function to achieve robust data fitting while rejecting outliers. Moreover, hard constraints, such as non-negativity or finite bounds on the variables, which are relevant for both tissue curve smoothing and sinogram or image smoothing, could be incorporated into the problem formulation. These extensions are of high interest to us and are also being pursued.

VI. Conclusions

We developed a direct (noniterative) algorithm for 1-D signal smoothing using quadratic data fitting with a penalty term based on the Huber function. The direct algorithm was derived from a dynamic programming reformulation of the objective function. We discussed in detail the solution method and computational cost. Two example applications, tissue concentration curve smoothing and sinogram smoothing, were used to demonstrate the effectiveness of the direct algorithm. Further applications such as using it as subproblem solvers in 2-D or 3-D image restoration and reconstruction were discussed, as well as possible extensions of the algorithm, such as including interval constraints.

Acknowledgments

We thank Michael Mozdy at University of Utah for proofreading the manuscript. The work of F. Noo was supported by Siemens Medical Solutions, USA; the concepts presented in this paper are based on research and are not commercially available.

References

1. Condat L. A Direct Algorithm for 1-D Total Variation Denoising. IEEE Signal Processing Letters. Nov.2013 20:1054–1057.
2. Johnson NA. A Dynamic Programming Algorithm for the Fused Lasso and L0-Segmentation. Journal of Computational and Graphical Statistics. Apr.2013 22:246–260.
3. Zhang X, Yu Y-L, Schuurmans D. Polar operators for structured sparse estimation. Advances in Neural Information Processing Systems. 2013:82–90.
4. Kolmogorov V, Pock T, Rolinek M. Total variation on a tree. SIAM Journal on Imaging Sciences. Jan.2016 9:605–636.
5. Condat L. A direct algorithm for 1d total variation denoising. 2012 hal-00675043v1.
6. Little KJ, La Rivière PJ. Sinogram restoration in computed tomography with an edge-preserving penalty. Medical Physics. 2015; 42(3):1307–1320. [PubMed: 25735286]
7. Gifford HC, Farncombe TH, King MA. Ga-67 tumor detection using penalized-em with nonanatomical regularizers. 2002 IEEE Nuclear Science Symposium Conference Record. Nov.2002 3:1397–1401.
8. Chlewicki W, Hermansen F, Hansen SB. Noise reduction and convergence of bayesian algorithms with blobs based on the huber function and median root prior. Physics in Medicine and Biology. 2004; 49(20):4717. [PubMed: 15566170]
9. Coakley KJ, Vecchia DF, Hussey DS, Jacobson DL. Neutron tomography of a fuel cell: Statistical learning implementation of a penalized likelihood method. IEEE Transactions on Nuclear Science. Oct.2013 60:3945–3954.
10. Wang AS, Stayman JW, Otake Y, Kleinszig G, Vogt S, Gallia GL, Khanna AJ, Siewerdsen JH. Soft-tissue imaging with C-arm cone-beam CT using statistical reconstruction. Physics in Medicine and Biology. 2014; 59(4):1005. [PubMed: 24504126]
11. Zhao B, Haldar JP, Christodoulou AG, Liang ZP. Image reconstruction from highly undersampled (k, t)-space data with joint partial separability and sparsity constraints. IEEE Transactions on Medical Imaging. Sep.2012 31:1809–1820. [PubMed: 22695345]
12. Lingala SG, Jacob M. Blind compressive sensing dynamic MRI. IEEE Transactions on Medical Imaging. Jun.2013 32:1132–1145. [PubMed: 23542951]
13. Defrise M, Vanhove C, Liu X. An algorithm for total variation regularization in high-dimensional linear problems. Inverse Problems. 2011; 27(6):065002.
14. Pock T, Cremers D, Bischof H, Chambolle A. Global solutions of variational models with convex regularization. SIAM Journal on Imaging Sciences. 2010; 3(4):1122–1145.

15. Papafitsoros K, Schönlieb CB. A combined first and second order variational approach for image reconstruction. Journal of Mathematical Imaging and Vision. 2014; 48(2):308–338.
16. Beck A, Teboulle M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM journal on imaging sciences. 2009; 2(1):183–202.
17. Nocedal, J., Wright, S. Numerical Optimization. 2. New York: Springer; Jul. 2006
18. Nesterov Y. Smooth minimization of non-smooth functions. Mathematical Programming. Dec.2004 103:127–152.
19. Nesterov Y. A method of solving a convex programming problem with convergence rate o (1/k2). Soviet Mathematics Doklady. 1983; 27:372–376.
20. Bellman R. The theory of dynamic programming. Bulletin of the American Mathematical Society. 1954; 60(6):503–515.
21. Dreyfus. The Art and Theory of Dynamic Programming. Academic Press; Jun. 1977
22. Rockafellar, RT., Wets, RJB. Variational Analysis. Springer; Berlin Heidelberg: 1998.
23. Boyd, S., Vandenberghe, L. Convex Optimization. 1. Cambridge University Press; Mar. 2004
24. Luo ZQ, Tseng P. On the convergence of the coordinate descent method for convex differentiable minimization. Journal of Optimization Theory and Applications. 72(1):7–35.
25. Wright SJ. Coordinate descent algorithms. Mathematical Programming. Mar.2015 151:3–34.
26. Burger, M., Osher, S. A Guide to the TV Zoo. Cham: Springer International Publishing; 2013. p. 1-70.
27. Parikh N, Boyd S, et al. Proximal algorithms. Foundations and Trends® in Optimization. 2014; 1(3):127–239.
28. Gill PE, Murray W, Saunders MA, Wright MH. Procedures for optimization problems with a mixture of bounds and general linear constraints. ACM Trans Math Softw. Aug.1984 10:282–298.
29. Schabel MC, Fluckiger JU, DiBella EVR. A model-constrained Monte Carlo method for blind arterial input function estimation in dynamic contrast-enhanced MRI: I. Simulations. Physics in Medicine and Biology. 2010; 55(16):4783. [PubMed: 20679691]
30. Wunderlich A, Noo F. Band-restricted estimation of noise variance in filtered backprojection reconstructions using repeated scans. IEEE Transactions on Medical Imaging. May.2010 29:1097–1113. [PubMed: 20236879]
31. Wunderlich A, Noo F. Confidence intervals for performance assessment of linear observers. Medical Physics. 2011; 38(S1):S57–S68. [PubMed: 21978118]
32. Chambolle A, Pock T. A first-order primal-dual algorithm for convex problems withapplications to imaging. Journal of Mathematical Imaging and Vision. 2011; 40(1):120–145.

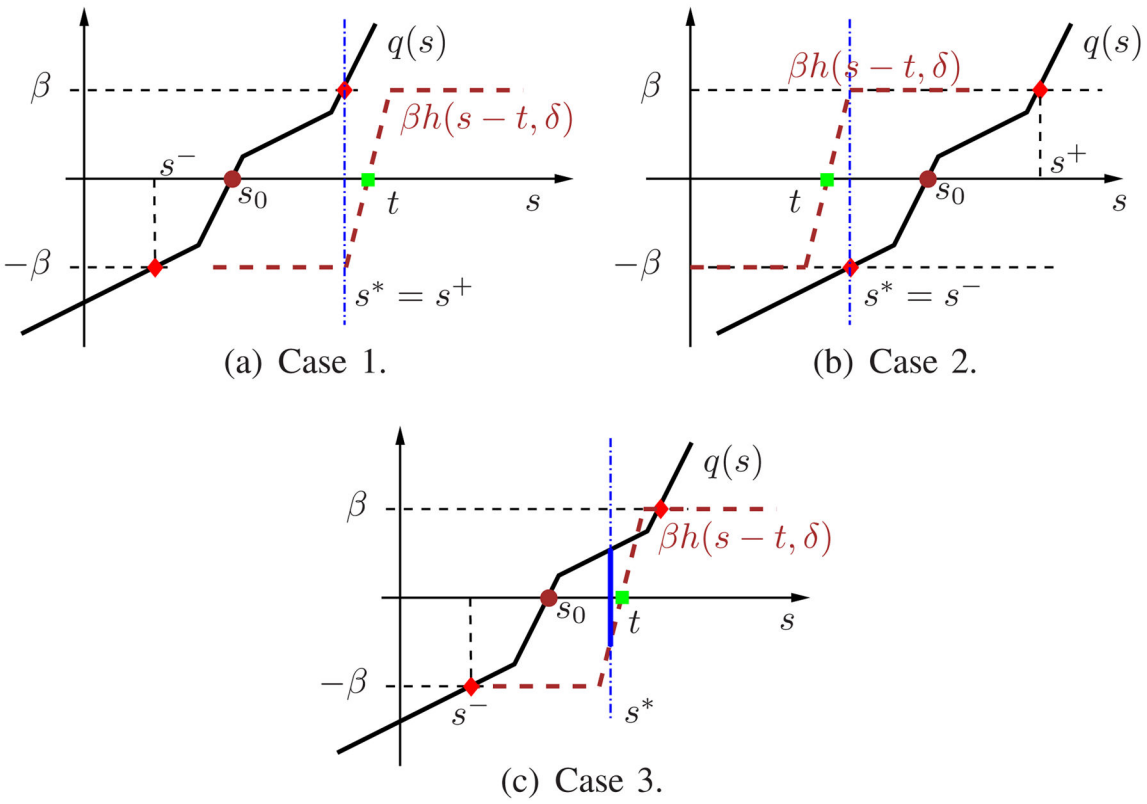

Fig. 1.

The three cases for the solution s^* as a function of t. The dashed (brown) ramp represents $\beta h(s - t, \delta)$. The solid piecewise linear function represents $q(s)$. The solution s^* (vertical dash-dot blue line) is such that $q(s^*) + \beta h(s^* - t, \delta) = 0$. The point s_0 is where $q(s_0) = 0$.

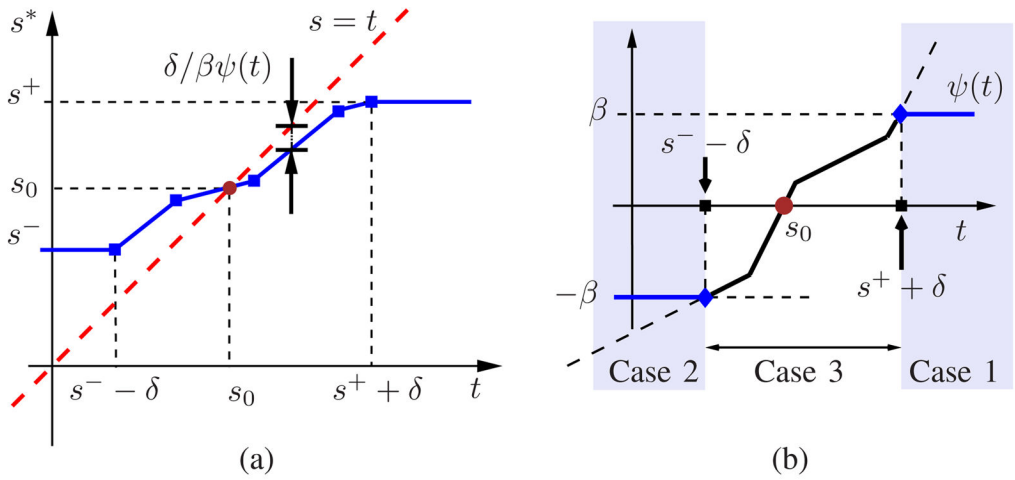

Fig. 2.

(a) The solution $s^*(t)$ is a piecewise linear, non-decreasing function of t. The vertical distance between the diagonal line $s = t$ and $s^*(t)$ is equal to $\delta q(s^*)/\beta$ or $\delta\psi(t)/\beta$. See (13) and (18). The point s_0 is such that $q(s_0) = 0$. As $\delta \to 0$, the distance between the diagonal line $s = t$ and the multi-linear section between $s^- - \delta$ and $s^+ + \delta$ would reduce to 0. (b) The function $\psi(t)$ is obtained from $q(s)$ by (i) clipping it at $(s^\pm, \pm\beta)$ and (ii) relabeling the axis; $\psi(t)$ is bounded between $\pm\beta$.

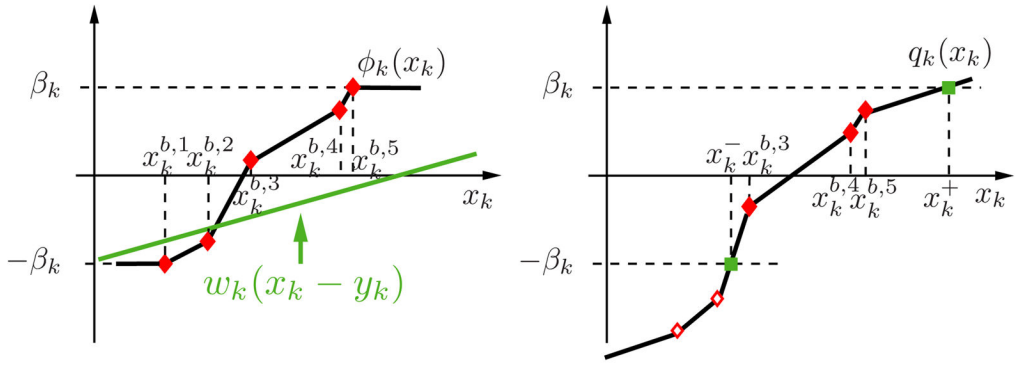

Fig. 3.

We assume that ϕ_k has five knots (filled red diamond markers). Depending on the data related term $w_k(x_k - y_k)$, some knots may be removed (trimmed) from $q_k(x_k)$ since we are only concerned in the range of x_k such that $|q_k(x_k)| \leq \beta_k$. In this example, the two knots at $x_k^{b,1}$ and $x_k^{b,2}$ are removed (open face red diamond markers); two new knots (filled green square markers) are inserted at the boundary $\pm\beta_k$.

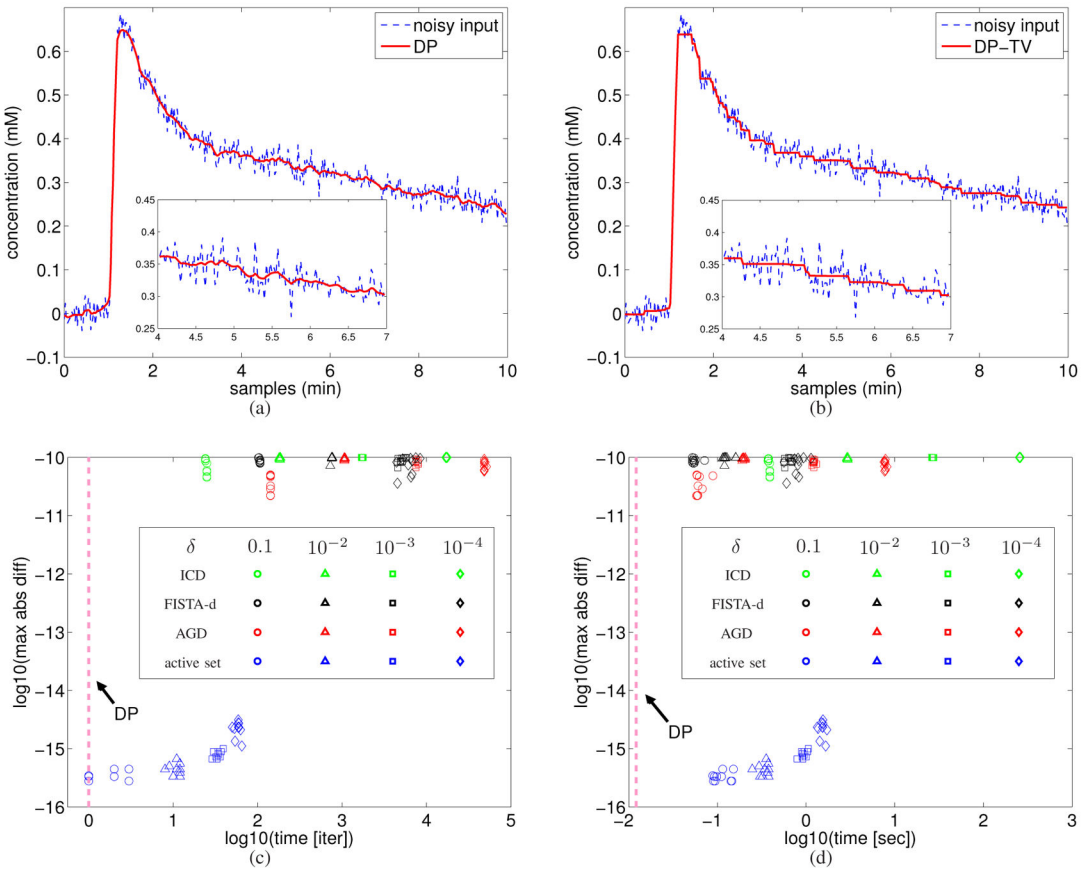

Fig. 4.

Tissue concentration curve smooching. (a) Huber smoothing, $\beta_k = 0.1$ and $\delta_k = 0.01$ for all $k = 1, \cdots N-1$. (b) TV smoothing ($\delta_k = 0$) generated a piecewise constant result, which is not physiologically reasonable. The insets in (a) and (b) zoom in to the time range [4, 7] min. (c) and (d) are comparisons of accuracy versus computation cost in terms of (c) iteration numbers and (d) wall clock time for the four methods, ICD, FISTA applied to the dual problem (FISTA-d), accelerated gradient descent (AGD), and Matlab's active set applied to the dual problem. The vertical axis, maximum absolute difference (ε), is measured with respect to the DP solution. The three methods, ICD, FISTA-d, and AGD, were stopped at $\varepsilon = 10^{-10}$, thus they all are located near the top of the figure. Each cluster of symbols represents 10 noise realizations. The dashed line in (c) is a symbolic representation of DP at one iteration; and the dashed line in (d) marks the average runtime of DP (0.012 sec).

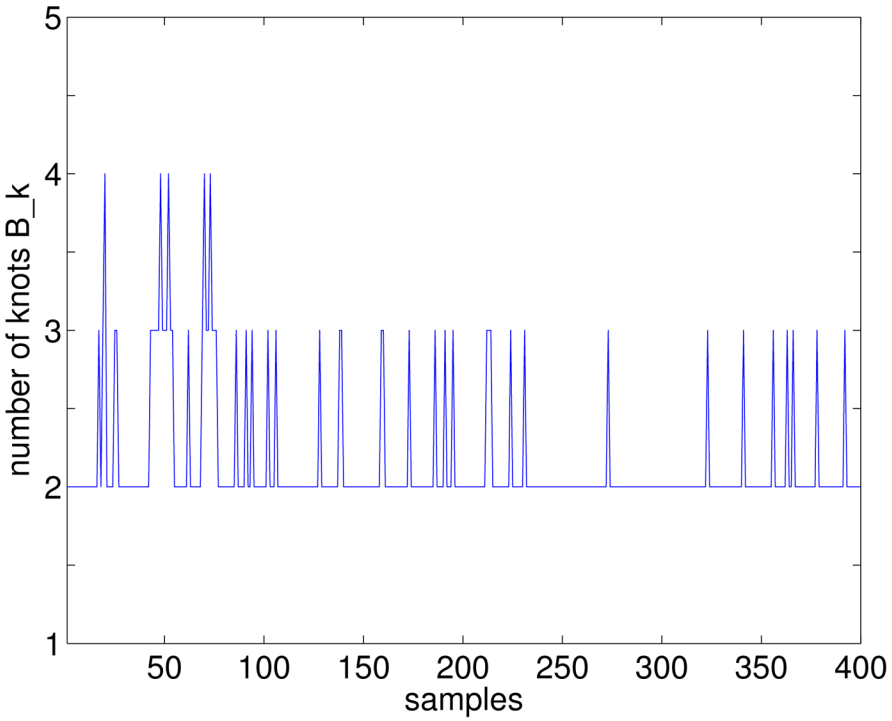

Fig. 5.

The number of knots B_k as a function of the sequence index k for the DP result in Fig. 4(a). The maximum number of knots is 4. In general, the number of knots depends on the values of δ and β.

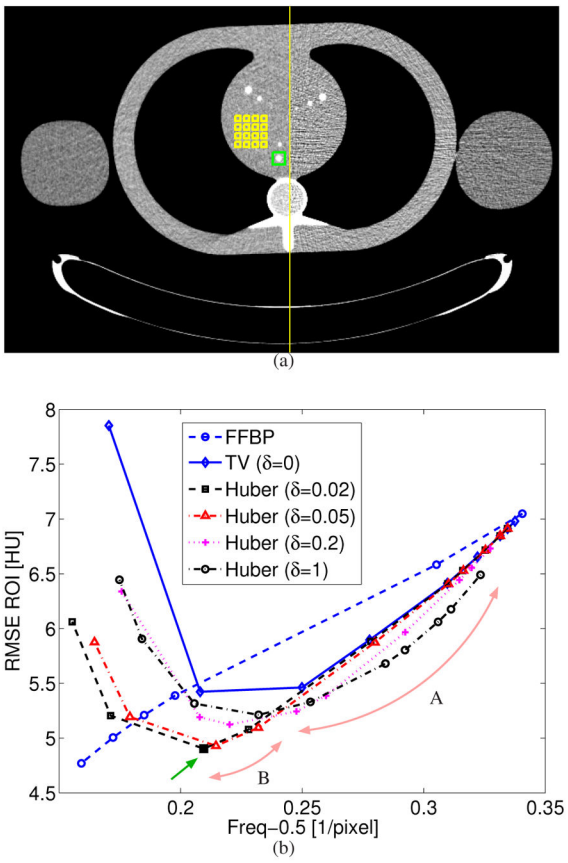

Fig. 6.

(a) A side-by-side comparison of FFBP reconstruction with (left) and without (right) sinogram smoothing. The horizontal streaks due to photon starvation are greatly reduced. The parameters used in sinogram smoothing were $\beta = 0.005$, $\delta = 0.02$. The yellow and green boxes mark the locations of the ROIs for RMSE and resolution calculation. (b) RMSE versus resolution tradeoffs for different values of δ in the Huber penalty and different apodization windows of FFBP. For each δ, the range of β includes the point with the lowest RMSE. The green arrow points to the parameter choice that produced the image on the left in (a). (C, W) = (0, 300) HU.

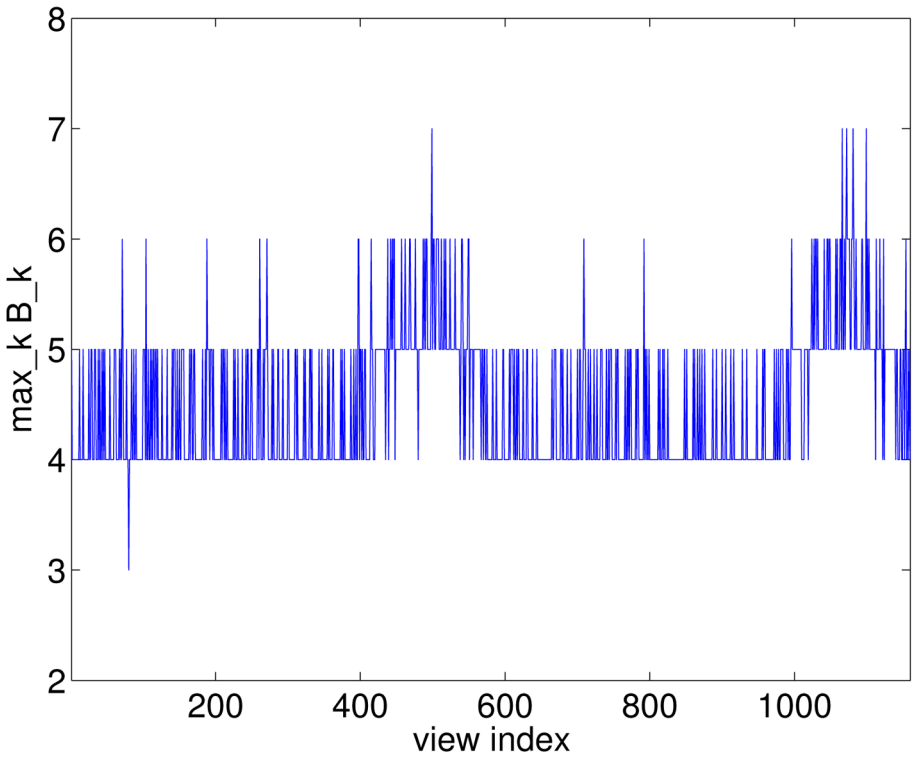

Fig. 7.

The maximum number of knots $\mathrm{MAX}_k B_k$ for each projection view when β= 0.005 and δ = 0.02.

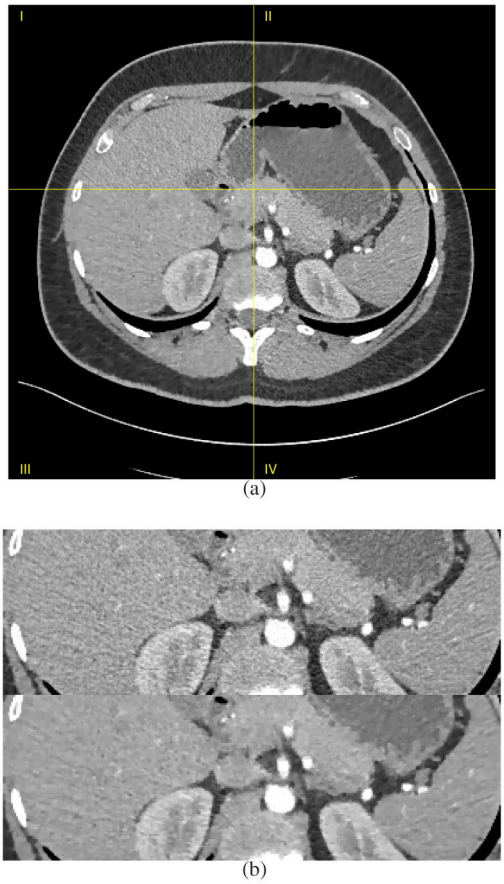

Fig. 8.

Image denoising. (a) A tessellation of the original (quadrants I, IV) and the denoised patient image (quadrants II, III). (b) The ROI around the liver, pancreas, and the kidney of the original (top) and the denoised image (bottom). (C, W) = (10, 400) HU.

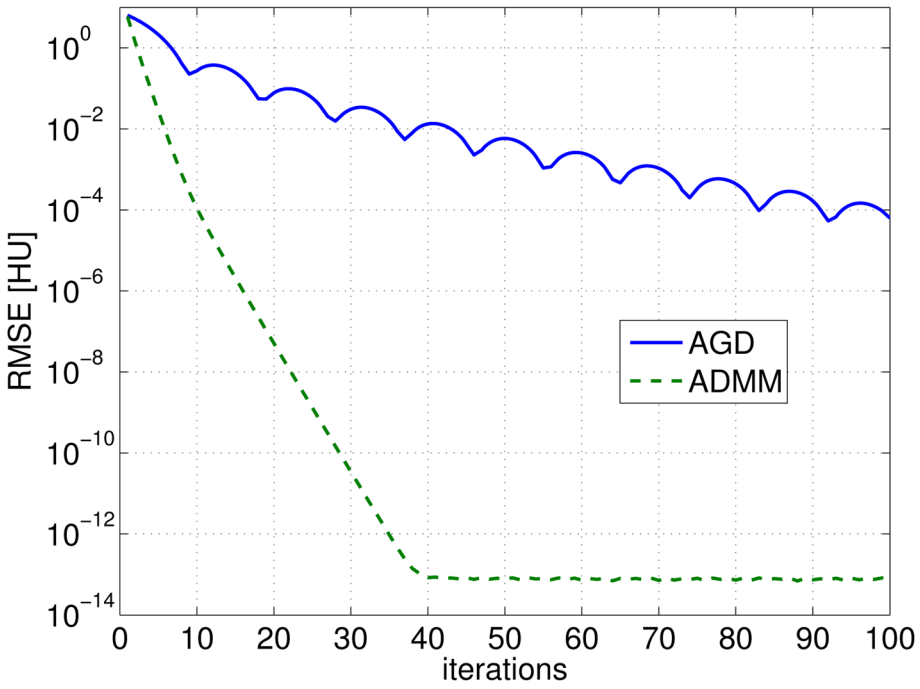

Fig. 9.

The root mean squared error (RMSE) in HU as a function of the iteration number for the accelerated gradient descent (AGD) and the proposed method outlined in (39).