TV denoising: An algorithm derived using majorization-minimization

Ivan Selesnick
selesi@poly.edu

September 10, 2012

1 Introduction

Total variation denoising (TVD) is an approach for noise reduction developed so as to preserve sharp edges in the underlying signal [13]. Unlike a conventional low-pass filter, TV denoising is defined in terms of an optimization problem. The output of the TV denoising ‘filter’ is obtained by minimizing a particular cost function. Any algorithm that solves the optimization problem can be used to implement TV denoising. However, it is not trivial because the TVD cost function is non-differentiable. Numerous algorithms have been developed to solve the TVD problem, e.g. [4–6,15,16].

Total variation is used not just for denoising, but for more general signal restoration problems, including deconvolution, interpolation, in-painting, compressed sensing, etc. [2]. In addition, the concept of total variation has been generalized and extended in various ways [3,10,12].

These notes describe an algorithm\(^1\) for TV denoising derived using the majorization-minimization (MM) approach, developed by Figueiredo et al. [9]. To keep it simple, these notes address TV denoising of 1-D signals only (ref. [9] considers 2D TV denoising for images). Interestingly, it is possible to obtain the exact solution to the TV denoising problem (for the 1-D case) without optimization, but instead using a direct algorithm based on a characterization of the solution. Recently, fast algorithm has been developed and C code made available [7].

Total variation denoising assumes that the noisy data \(y(n)\) is of the form

\[
y(n) = x(n) + w(n), \quad n = 0, \ldots, N - 1
\]

where \(x(n)\) is a (approximately) piecewise constant signal and \(w(n)\) is white Gaussian noise. TV denoising estimates the signal \(x(n)\) by solving the optimization problem:

\[
\arg\min_x \sum_{n=0}^{N-1} |y(n) - x(n)|^2 + \lambda \sum_{n=1}^{N-1} |x(n) - x(n-1)|.
\]

The regularization parameter \(\lambda > 0\) controls the degree of smoothing. Increasing \(\lambda\) gives more weight to the second term which measures the fluctuation of the signal \(x(n)\).

\(^1\)MATLAB software online: http://eeweb.poly.edu/iselesni/teaching/lecture_notes/TVDmm/index.html.
1.1 Notation

The $N$-point signal $x$ is represented by the vector
\[ x = [x(0), \ldots, x(N-1)]^t. \]

The $\ell_1$ norm of a vector $v$ is defined as
\[ \|v\|_1 = \sum_n |v(n)|. \]

The $\ell_2$ norm of a vector $v$ is defined as
\[ \|v\|_2 = \left[ \sum_n |v(n)|^2 \right]^{\frac{1}{2}}. \]

The matrix $D$ is defined as
\[
D = \begin{bmatrix}
-1 & 1 & & \\
 & -1 & 1 & \\
& & & \ddots \\
& & & -1 & 1 \\
\end{bmatrix}.
\]

The first-order difference of an $N$-point signal $x$ is given by $Dx$ where $D$ is of size $(N-1) \times N$.

Note, for later, that $DD^t$ is a tridiagonal matrix of the form:
\[
DD^t = \begin{bmatrix}
2 & -1 & & \\
-1 & 2 & -1 & \\
& & \ddots & \\
& & & -1 & 2 & -1 \\
\end{bmatrix}.
\]

The total variation of the $N$-point signal $x(n)$ is given by
\[ \text{TV}(x) := \|Dx\|_1 = \sum_{n=1}^{N-1} |x(n) - x(n-1)|. \]

With this notation, the TV denoising problem (2) can be written compactly as
\[
\arg\min_{x \in \mathbb{R}^N} \|y - x\|_2^2 + \lambda \|Dx\|_1.
\]

2 Majorization-Minimization

Majorization-minimization (MM) is an approach to solve optimization problems that are too difficult to solve directly. Instead of minimizing the cost function $F(x)$ directly, the MM approach solves a sequence of optimization problems, $G_k(x)$, $k = 0, 1, 2, \ldots$. The idea is that each $G_k(x)$ is easier to solve that $F(x)$. The MM approach produces a sequence $x_k$, each being obtained by
minimizing $G_{k-1}(x)$. To use MM, one must specify the functions $G_k(x)$. The trick is to choose the $G_k(x)$ so that they are easy to solve, but they should also each approximate $F(x)$.

The MM approach requires that each function $G_k(x)$ is a majorizer of $F(x)$, i.e.,

$$G_k(x) \geq F(x), \quad \forall x$$

and that it agrees with $F(x)$ at $x_k$,

$$G_k(x_k) = F(x_k).$$

In addition, $G_k(x)$ should be convex functions. The MM approach then obtains $x_{k+1}$ by minimizing $G_k(x)$.

Figure 1 illustrates the MM procedure with a simple example. For clarity, the figure illustrates the minimization of a univariate function. However, the MM procedure works in the same way for the minimization of multivariate functions, and it is in the multivariate case where the MM procedure is especially useful.

The majorization-minimization approach to minimize the function $F(x)$ can be summarized as:

1. Set $k = 0$. Initialize $x_0$.
2. Choose $G_k(x)$ such that
   (a) $G_k(x) \geq F(x)$ for all $x$
   (b) $G_k(x_k) = F(x_k)$
3. Set $x_{k+1}$ as the minimizer of $G_k(x)$.
   $$x_{k+1} = \arg\min_x G_k(x)$$ (6)
4. Set $k = k + 1$ and go to step (2.)

When $F(x)$ is convex, then under mild conditions, the sequence $x_k$ produced by MM converges to the minimizer of $F(x)$. More details about the majorization-minimization procedure can be found in [8] and references therein.

**Example majorizer.** An upper bound (majorizer) of $f(t) = |t|$ that agrees with $f(t)$ at $t = t_k$ is

$$g(t) = \frac{1}{2|t_k|} t^2 + \frac{1}{2}|t_k|$$ (7)

as illustrated in Fig. 2. The figure makes clear that

$$g(t) \geq f(t), \quad \forall t$$

$$g(t_k) = f(t_k)$$
(a) Function $F(x)$ to be minimized. MM is initialized with $x_0$.

(b) Iteration 1. Majorizor $G_0(x)$ coincides with $F(x)$ at $x_0$. Minimize $G_0(x)$ to get $x_1$.

(c) Iteration 2. Majorizor $G_1(x)$ coincides with $F(x)$ at $x_1$. Minimize $G_1(x)$ to get $x_2$.

The $x_k$ converge to the minimizer of $F(x)$.

Figure 1: Illustration of majorization-minimization (MM) procedure.
(a) Cost function $F(x)$ to be minimized; and initialization, $x_0$.
(b) Iteration 1. Majorizor $G_0(x)$ coincides with $F(x)$ at $x_0$. Minimize $G_0(x)$ to get $x_1$.
(c) Iteration 2. Majorizor $G_1(x)$ coincides with $F(x)$ at $x_1$. Minimize $G_1(x)$ to get $x_2$.

The $x_k$ converge to the minimizer of $F(x)$.
3 TV Denoising Algorithm

One way to apply MM to TV denoising is to majorize TV(x) by a quadratic function of x, as described in ref. [9]. Then the TVD cost function $F(x)$ can be majorized by a quadratic function, which can in turn be minimized by solving a system of linear equations.

To that end, using (7), we can write

$$\frac{1}{2} |t_k|^2 + \frac{1}{2} |t_k| \geq |t| \quad \forall t \in \mathbb{R}$$

Using $v(n)$ for $t$ and summing over $n$ gives

$$\sum_n \left[ \frac{1}{2} |v_k(n)| v^2(n) + \frac{1}{2} |v_k(n)| \right] \geq \sum_n |v(n)|$$

which can be written compactly as

$$\frac{1}{2} v^T \Lambda_k v + \frac{1}{2} \|v_k\|_1 \geq \|v\|_1$$

where $\Lambda_k$ is the diagonal matrix

$$\Lambda_k := \begin{bmatrix} \frac{1}{|v_k(1)|} & & & \\ & \frac{1}{|v_k(2)|} & & \\ & & \ddots & \\ & & & \frac{1}{|v_k(N)|} \end{bmatrix} = \text{diag}(1./|v_k|).$$

The notation $1./v$ denotes element-wise division\(^2\) and the absolute value is applied element-wise.

\(^2\)The notation $./$ is used in MATLAB.
Using $Dx$ for $v$, we can write
\[
\frac{1}{2} x^t D^t \Lambda_k D x + \frac{1}{2} \|Dx_k\|_1 \geq \|Dx\|_1
\]  
(8)

where
\[
\Lambda_k := \text{diag}(1./|Dx|).
\]

Note in (8) that the majorizor of $\|Dx\|_1$ is a quadratic function of $x$. Also note that the term $\|Dx_k\|_1$ in (8) should be considered a constant — it is fixed as $x_k$ is the value of $x$ at the previous iteration. Similarly, $\Lambda_k$ in (8) is also not a function of $x$.

A majorizor of the TV cost function (5) can be obtained from (8) by adding $\|y-x\|_2^2$ to both sides,
\[
\|y-x\|_2^2 + \frac{1}{2} x^t D^t \Lambda_k D x + \lambda \frac{1}{2} \|Dx_k\|_1 \geq \|y-x\|_2^2 + \lambda \|Dx\|_1.
\]

Therefore a majorizor $G_k(x)$ for the TV cost function is given by
\[
G_k(x) = \|y-x\|_2^2 + \lambda \frac{1}{2} x^t D^t \Lambda_k D x + \lambda \frac{1}{2} \|Dx_k\|_1, \quad \Lambda_k = \text{diag}(1./|Dx|)
\]
which satisfies $G_k(x_k) = F(x_k)$ by design. Using MM, we obtain $x_k$ by minimizing $G_k(x)$,
\[
x_{k+1} = \text{argmin}_x \|y-x\|_2^2 + \lambda \frac{1}{2} x^t D^t \Lambda_k D x + \lambda \frac{1}{2} \|Dx_k\|_1.
\]  
(9)

An explicit solution to (9) is given by
\[
x_{k+1} = \left( I + \frac{\lambda}{2} D^t \Lambda_k D \right)^{-1} y.
\]  
(10)

A problem with update (10) is that as the iterations progress, some values of $Dx_k$ will generally go to zero, and therefore some entries of $\Lambda_k$ in (10) will go to infinity. This issue is addressed in Ref. [9] by rewriting the equation using the matrix inverse lemma. By the matrix inverse lemma,
\[
\left( I + \frac{\lambda}{2} D^t \Lambda_k D \right)^{-1} = I - D^t \left( \frac{2}{\lambda} \Lambda_k^{-1} + DD^t \right)^{-1} D
\]
where
\[
\Lambda_k^{-1} = \text{diag}(|Dx_k|).
\]

Now the update equation (10) becomes
\[
x_{k+1} = y - D^t \left( \frac{2}{\lambda} \text{diag}(|Dx_k|) + DD^t \right)^{-1} Dy.
\]  
(11)

Observe that even if some elements of $Dx_k$ are zero, no division by zero arises in (11).

The update (11) calls for the solution to a linear system of equations. In general, it is desirable to avoid such a computation in an iterative filtering algorithm due to the high computational
cost of solving linear systems (especially when the signal $y$ is very long and the system is very large). However, the matrix $\left[ \frac{2}{\lambda} \text{diag}(|Dx_k|) + DD^t \right]$ in (11) is a sparse banded matrix; it consists of only three diagonals — the main diagonal, one upper diagonal, and one lower diagonal. This is because $DD^t$ is tridiagonal as shown in (4). Therefore, the linear system in (11) can be solved very efficiently [11, Sect 2.4]. Further, the whole matrix need not be stored in memory, only the three diagonals.

A MATLAB function `TVD_mm` listed as Program 1 implements TV denoising based on the update (11). The function uses the sparse matrix structure in MATLAB so as to avoid high memory requirements and so as to invoke sparse banded system solvers. MATLAB uses LAPACK [1] to solve the sparse banded system in the program `TVD_mm`. The algorithm used by MATLAB to solve a sparse linear system can be monitored using the command `spparms('spumoni',3)`.

An example of TV denoising is shown in Fig. 3. The history of the cost function through the progression of the algorithm is shown in the figure. It can be seen that after 20 iterations the cost function has leveled out, suggesting that the algorithm has practically converged.

Another algorithm for 1-D TV denoising is Chambolle’s algorithm [5], a variant of which is the ‘iterative clipping’ algorithm [14]. This algorithm is computationally simpler than the MM algorithm because it does not call for the solution to a linear system at each iteration. However, it may converge slowly. For the denoising problem illustrated in Fig. 3, the convergence of both the iterative clipping and MM algorithms are shown in Fig. 4. It can be seen that the MM algorithm converges in fewer iterations.
function [x, cost] = TVD_mm(y, lam, Nit)
% [x, cost] = TVD_mm(y, lam, Nit)
% Total variation denoising using majorization-minimization
% and a sparse linear system solver.
%
% INPUT
%  y - noisy signal
%  lam - regularization parameter
%  Nit - number of iterations
%
% OUTPUT
%  x - denoised signal
%  cost - cost function history
%
% Reference
% 'On total-variation denoising: A new majorization-minimization
% algorithm and an experimental comparison with wavelet denoising.'

% Ivan Selesnick
% selesi@poly.edu
% 2011

y = y(:); % ensure column vector
cost = zeros(1, Nit); % cost function history
N = length(y);

e = ones(N-1, 1);
DDT = spdiags([-e 2*e -e], [-1 0 1], N-1, N-1); % D*D' (sparse matrix)
D = @(x) diff(x); % D (operator)
DT = @(x) [-x(1); -diff(x); x(end)]; % D'

x = y; % initialization
Dx = D(x);

for k = 1:Nit
    F = 2/lam * spdiags(abs(Dx), 0, N-1, N-1) + DDT; % F : sparse matrix
    % F = 2/lam * diag(abs(D(x))) + DDT;
    x = y - DT(F\D(y)); % Solve sparse linear system
    Dx = D(x);
    cost(k) = sum(abs(x-y).^2) + lam * sum(abs(Dx)); % keep cost function history
end

Program 1: MATLAB program for TV denoising using majorization-minimization. The program is based on update (11).
4 Conclusion

Total variation denoising is a method to smooth signals based on the minimization of a non-differentiable cost function. Unlike a conventional low-pass filter, the output of the TV denoising ‘filter’ can only be obtained through a numerical algorithm. Total variation denoising is most appropriate for piecewise constant signals, however, it has been modified and extended so as to be effective for more general signals.

5 Exercises

1. Reproduce figures like those of the example (using a ‘blocky’ signal). Try different values of $\lambda$. How does the solution change as $\lambda$ is increased or decreased?

2. Compare TV denoising with low-pass filtering (e.g. Butterworth filter, or FIR filters, etc). Apply each method to the same signal. Plot the denoised/filtered signals using each method and discuss the differences you observe.

3. Perform TV denoising on a signal that is not ‘blocky’ (which has slopes or oscillatory behavior). You should see ‘stair-case’ artifacts in the denoised signal. Show these artifacts in a figure and explain why they arise.

4. Is TV denoising linear? (Conventional low-pass filters, e.g. FIR or IIR, are linear.) Illustrate that TV denoising satisfies (or does not) the superposition property by performing TV denoising on each of two signals and their sum.

5. Show the solution to problem (9) is given by (10).
6. For denoising a noisy signal using TV denoising, devise a method or formula to set the regularization parameter $\lambda$. You can assume that the variance $\sigma^2$ of the noisy signal is known. Show examples of your method.

7. Explain why $D^t$ can be implemented in MATLAB by the command

$$DT = \emptyset(x) [-x(1); -\text{diff}(x); x(\text{end})];$$

8. Modify Program 1 so that the matrix $F$ is not sparse. Measure the run-times of the original and modified programs. Is the sparse version faster? Use a long signal and many iterations to see the difference more clearly.

9. Second-order TV denoising is based on the second-order difference instead of the first-order difference. Modify the algorithm and MATLAB program so that it performs second-order TV denoising. Compare first and second order TV denoising using ‘blocky’ and non-‘blocky’ signals, and comment on your observations.

References


