Application of Adjoint Operators in Gradient Computations

James Folberth
Advisor: Stephen Becker

University of Colorado at Boulder

5 March, 2016

Outline

1. A generic optimiztion problem
2. Example 1: Image Deblurring
3. Example 2: Blind Channel Estimation

A Generic Problem

Consider

$$\min_x \frac{1}{2}\|\mathcal{A}x - b\|_2^2 + \lambda\|x\|_1.$$

- $\mathcal{A}$ is a linear operator on problem variable x.
- b is measured data (e.g. blurry image).
- Include $\lambda\|x\|_1$, to induce sparsity in x (hopefully).

Define

$$f(x) = \frac{1}{2}\|\mathcal{A}x - b\|_2^2, \quad g(x) = \lambda\|x\|_1.$$

$f(x)$ is convex, differentiable, $g(x)$ is convex, non-differentiable.

$$\nabla f(x) = \mathcal{A}^*(\mathcal{A}x - b).$$

Proximal Gradient Method

$$\min_x \frac{1}{2}\|\mathcal{A}x - b\|_2^2 + \lambda\|x\|_1.$$

$$\nabla f(x) = \mathcal{A}^* \left(\mathcal{A}x - b\right), \quad x^+ = \mathrm{prox}_{tg}\left(x - t\nabla f(x)\right).$$

Need to efficiently compute

- $\mathrm{prox}_{tg}(x)$ with $g(x) = \lambda\|x\|_1$. “Shrinkage” is fast.
- $\mathcal{A}$. Usually have fast forward and inverse transform (e.g. FFT, discrete wavelet transform).
- $\mathcal{A}^*$. Sometimes not so easy... Let's look at a couple examples.

Image Deblurring Problem

Observation: natural images tend to have sparse wavelet coefficients.

- b - observed blurred image, with known blurring operator $\mathcal{R}$ (e.g. Gaussian PSF applied efficiently in Fourier domain)
- $\mathcal{W}$ - multi-level wavelet synthesis operator
- x - wavelet coefficients

Natural problem formulation is

$$\min_x \frac{1}{2}\|\mathcal{R}\mathcal{W}x - b\|_2^2 + \lambda\|x\|_1.$$

$$\nabla f(x) = \mathcal{W}^*\mathcal{R}^*\left(\mathcal{R}\mathcal{W}x - b\right).$$

Image Deblurring Problem

Observed image:

Recovered image:

Adjoint of Wavelet Operator

$$\nabla f(x) = \mathcal{W}^* \mathcal{R}^* \left(\mathcal{R} \mathcal{W} x - b \right).$$

- $\mathcal{W}$ is wavelet synthesis (reconstruction). Standard routine in libraries.
- $\mathcal{R}$ and $\mathcal{R}^*$ for blurring PSF can be applied rapidly in Fourier domain (FFT).
- What about $\mathcal{W}^*$? Not a standard operation like $\mathcal{W}$ and $\mathcal{W}^\dagger$.

If $\mathcal{W}$ is orthogonal, $\mathcal{W}^* = \mathcal{W}^\dagger$.
If $\mathcal{W}$ is biorthogonal, $\mathcal{W}^* \approx \mathcal{W}^\dagger$.
So, one option is

$$\nabla f(x) \approx \mathcal{W}^\dagger \mathcal{R}^* \left(\mathcal{R} \mathcal{W} x - b \right).$$

Adjoint of Wavelet Operator

Digging around in frame theory a bit, it turns out $\mathcal{W}^* = \tilde{\mathcal{W}}^\dagger$: the adjoint of wavelet synthesis is dual wavelet analysis.

But we must also handle boundary conditions. This is usually done by extending the signal via $\mathcal{E}$ to satisfy the BCs.

The relation $\mathcal{W}^* = \tilde{\mathcal{W}}^\dagger$ holds for $\mathcal{E}$ being zero-padding, since $\mathcal{E}_{\text{zpd}}^* = \mathcal{E}_{\text{zpd}}^\dagger$. Let $\mathcal{W}_{\text{zpd}}$ be wavelet synthesis with zero BCs.

For general $\mathcal{E}$, we have

$$\mathcal{W}^\dagger = \mathcal{W}_{\text{zpd}}^\dagger \mathcal{E} \implies \mathcal{W} = \mathcal{E}^\dagger \mathcal{W}_{\text{zpd}} \implies \mathcal{W}^* = \mathcal{W}_{\text{zpd}}^* (\mathcal{E}^\dagger)^*$$

and

$$\mathcal{W}^* = \tilde{\mathcal{W}}_{\text{zpd}}^\dagger (\mathcal{E}^\dagger)^*.$$

Adjoint of Pseudoinverse Extension

Now we just need to implement $(\mathcal{E}^\dagger)^*$! $\tilde{\mathcal{W}}^\dagger_{\mathrm{zpd}}$ is a standard and fast operation.

Consider a signal $y[n]$, $n = 0, \ldots, N-1$. Let L_p be the length of wavelet analysis filters.

Zero padding:

$$\underbrace{0, \ldots, 0}_{L_p - 1}, y[0], \ldots, y[N-1], \underbrace{0, \ldots, 0}_{L_p - 1}.$$

Half-point symmetric:

$$\underbrace{y[L_p - 1], \ldots, y[0]}_{\text{Left extension}}, y[0], \ldots, y[N-1], \underbrace{y[N-1], \ldots, y[N + L_p - 2]}_{\text{Right extension}}.$$

Zero padding

Zero padding as a linear operator:

$$\mathcal{E}_{\text{zpd}} = \begin{bmatrix} 0_{(L_p-1)\times N} \\ I_{N\times N} \\ 0_{(L_p-1)\times N} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

In this case

$$\left(\mathcal{E}^{\dagger}_{\text{zpd}}\right)^{*} = \mathcal{E}_{\text{zpd}}.$$

This is what allows us the factorization

$$\mathcal{W}^{*} = \tilde{\mathcal{W}}^{\dagger}_{\text{zpd}} \left(\mathcal{E}^{\dagger}\right)^{*}.$$

Half-point symmetric

Half-point symmetric extension as a linear operator:

$$
\mathcal{E}_{\mathsf{sym}} = \begin{bmatrix}
 & & \iddots & & \\
 & 1 & & & \\
1 & & & & \\
1 & & & & \\
 & 1 & & & \\
 & & \ddots & & \\
 & & & 1 & \\
 & & & & 1 \\
 & & & & 1 \\
 & & & 1 & \\
 & & \iddots & &
\end{bmatrix}, \quad \left(\mathcal{E}_{\mathsf{sym}}^{\dagger}\right)^{*} = \begin{bmatrix}
 & \iddots & & & & & & & \\
1/2 & & & & & & & & \\
1/2 & & & & & & & & \\
 & \ddots & & & & & & & \\
 & & 1/2 & & & & & & \\
 & & & 1 & & & & & \\
 & & & & \ddots & & & & \\
 & & & & & 1 & & & \\
 & & & & & & 1/2 & & \\
 & & & & & & & \ddots & \\
 & & & & & & & & 1/2 \\
 & & & & & & & & 1/2 \\
 & & & & & & & \iddots &
\end{bmatrix}.
$$

Image Deblurring Problem

200 iterations of FISTA, $\mathcal{W}^\dagger$ is a 3-stage CDF 9/7 wavelet transform, $\lambda = 2 \times 10^{-5}$

Using $\mathcal{W}^* \approx \mathcal{W}^\dagger$:

$$\frac{\|\mathcal{W}x - y\|_2}{\|y\|_2} = 7.25 \times 10^{-2}$$

Using $\mathcal{W}^* = \tilde{\mathcal{W}}^\dagger_{\text{zpd}}(\mathcal{E}^\dagger)^*$:

$$\frac{\|\mathcal{W}x - y\|_2}{\|y\|_2} = 7.24 \times 10^{-2}$$

Image Deblurring Problem

- $\mathcal{W}^* = \tilde{\mathcal{W}}_{\mathsf{zpd}}^\dagger (\mathcal{E}^\dagger)^*$.
- $\tilde{\mathcal{W}}_{\mathsf{zpd}}^\dagger$ is standard in wavelet libraries.
- $(\mathcal{E}^\dagger)^*$ is closed-form (once you find it) and fast.
- So we can apply $\mathcal{W}^*$ efficiently *and correctly* in

$$\nabla f(x) = \mathcal{W}^* \mathcal{R}^* \left(\mathcal{R}\mathcal{W}x - b \right).$$

BCE Problem

Unknown source sends signal s over unknown channels with impulse responses h_i. We observe channel outputs

$$x_i[n] = \{h_i * s\}[n].$$

Can we recover source and channel IRs?
Let's restrict ourselves to h_i and s real-valued.
Notice that for any $\alpha \neq 0$,

$$x_i[n] = \{h_i * s\}[n] = \{\alpha h_i * \frac{1}{\alpha} s\}[n].$$

So we can maybe recover h_i and s up to a factor.

BCE Problem

For simplicity of notation, consider a single channel h with output x. Let h be of length K and s be of length N; x will be of length $K + N - 1$.

One can write the convolution as linear operator on the $K \times N$ matrix hs^T:

$$x = h * s = \mathcal{A}(hs^T).$$

Assuming h and s should be sparse in time, a natural problem formulation is

$$\min_{h,s} \frac{1}{2}\|\mathcal{A}(hs^T) - x\|_2^2 + \lambda_h\|h\|_1 + \lambda_s\|s\|_1.$$

This is non-convex. Can use other regularization terms (e.g. $\|h\|_{\mathrm{TV}}$).

Adjoint of $\mathcal{A}$

Define $f(h, s) = 1/2\|\mathcal{A}(hs^T) - x\|_2^2$.

The required gradients are

$$\nabla_h f(h, s) = \left[\mathcal{A}^*(\mathcal{A}(hs^T) - x)\right] s$$
$$\nabla_s f(h, s) = \left[\mathcal{A}^*(\mathcal{A}(hs^T) - x)\right]^T h.$$

Note that $\mathcal{A}$ takes a matrix and returns a vector. So $\mathcal{A}^*$ must take a vector and return a matrix (of appropriate size).

Adjoint of $\mathcal{A}$

We know the action of $\mathcal{A}(hs^T)$:

$$x[n] = \sum_{k=k_1(n)}^{k_2(n)} h[k]s[n-k],$$

where $k_1(n) = \max\{0, n+1-N\}$ and $k_2(n) = \min\{K-1, n\}$.

Adjoint of $\mathcal{A}$

We know the action of $\mathcal{A}(hs^T)$:

$$
\begin{array}{ccccc}
 & h[0]s[0] & h[0]s[1] & h[0]s[2] & h[0]s[3] \\
x[0] & & & & \\
 & h[1]s[0] & h[1]s[1] & h[1]s[2] & h[1]s[3] \\
x[1] & & & & \\
 & h[2]s[0] & h[2]s[1] & h[2]s[2] & h[2]s[3] \\
x[2] & & & & \\
 & h[3]s[0] & h[3]s[1] & h[3]s[2] & h[3]s[3] \\
x[3] & & & & \\
 & h[4]s[0] & h[4]s[1] & h[4]s[2] & h[4]s[3] \\
x[4] & & & & \\
 & h[5]s[0] & h[5]s[1] & h[5]s[2] & h[5]s[3] \\
x[5] & & & & \\
 & h[6]s[0] & h[6]s[1] & h[6]s[2] & h[6]s[3] \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{array}
$$

Adjoint of $\mathcal{A}$

Adjoint is defined via

$$\langle \mathcal{A}(X), y \rangle = \langle X, \mathcal{A}^*(y) \rangle \quad \forall X \forall y.$$

Plug in explicit form of $\mathcal{A}(X)$:

$$\begin{aligned}
\langle \mathcal{A}(X), y \rangle &= \sum_{n=0}^{K+N-2} y[n] \left(\sum_{k=k_1(n)}^{k_2(n)} X[k, n-k] \right) \\
&= y[0]X[0,0] + y[1]\left(X[0,1] + X[1,0]\right) \\
&\quad + y[2]\left(X[0,2] + X[1,1] + X[2,0]\right) + \cdots .
\end{aligned}$$

Notice that $X[i,j]$ is always multiplied by $y[i+j]$. Defines Hankel matrix!

Adjoint of $\mathcal{A}$

$$
\begin{aligned}
\langle \mathcal{A}(X), y \rangle &= \sum_{n=0}^{K+N-2} y[n] \left(\sum_{k=k_1(n)}^{k_2(n)} X[k, n-k] \right) \\
&= \sum_{i=0}^{K-1} \sum_{j=0}^{N-1} X[i,j] y[i+j] \\
&= \sum_{i=0}^{K-1} \sum_{j=0}^{N-1} X[i,j] Y[i,j] \\
&= \langle X, Y \rangle = \langle X, \mathcal{A}^*(y) \rangle \,,
\end{aligned}
$$

where we define the $K \times N$ Hankel matrix Y by $Y[i,j] = y[i+j]$ so

$$\mathcal{A}^*(y) = Y.$$

Hankel matrix-vector product

The Hankel matrix Y is dense but structured:

$$Y = \begin{bmatrix} y[0] & y[1] & y[2] & y[3] & y[4] \\ y[1] & y[2] & y[3] & y[4] & y[5] \\ y[2] & y[3] & y[4] & y[5] & y[6] \end{bmatrix}.$$

Reorder columns to get Toeplitz matrix:

$$T = YP = \begin{bmatrix} y[4] & y[3] & y[2] & y[1] & y[0] \\ y[5] & y[4] & y[3] & y[2] & y[1] \\ y[6] & y[5] & y[4] & y[3] & y[2] \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Hankel matrix-vector product

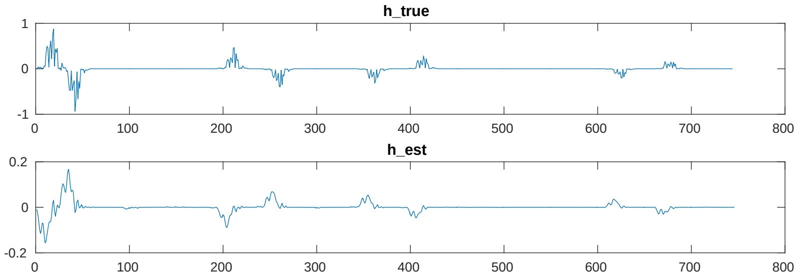
Embed Toeplitz matrix in circulant matrix:

$$C = \left[\begin{array}{ccccc|cc} y[4] & y[3] & y[2] & y[1] & y[0] & y[6] & y[5] \\ y[5] & y[4] & y[3] & y[2] & y[1] & y[0] & y[6] \\ y[6] & y[5] & y[4] & y[3] & y[2] & y[1] & y[0] \\ \hline y[0] & y[6] & y[5] & y[4] & y[3] & y[2] & y[1] \\ y[1] & y[0] & y[6] & y[5] & y[4] & y[3] & y[2] \\ y[2] & y[1] & y[0] & y[6] & y[5] & y[4] & y[3] \\ y[3] & y[2] & y[1] & y[0] & y[6] & y[5] & y[4] \end{array}\right].$$

We can do a fast circulant mat-vec via the FFT! Thus, we can compute and apply $\mathcal{A}^*(y)$ rapidly.

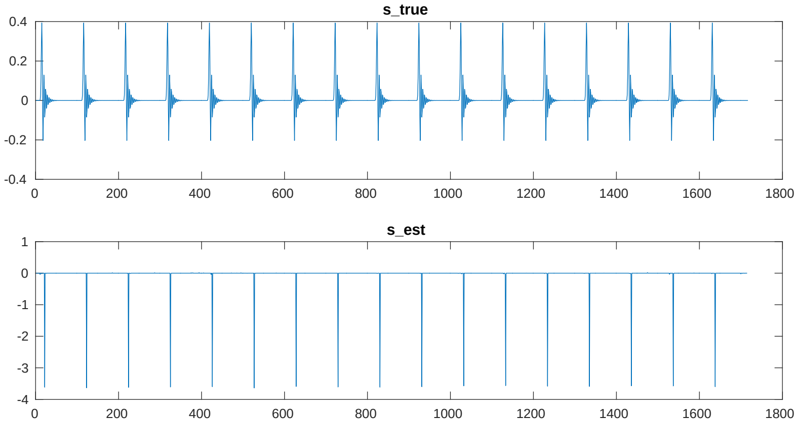
Example estimation

Still a work in progress!



Example estimation

Still a work in progress!



Thanks!

We can now compute the adjoint wavelet transform:

$$\mathcal{W}^* = \tilde{\mathcal{W}}^{\dagger}_{\mathsf{zpd}}(\mathcal{E}^{\dagger})^*.$$

Interesting adjoint in blind channel estimation problem:

$$x = h * s = \mathcal{A}(hs^T).$$

References:

- A. Beck, M. Teboulle, *A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems*, SIAM Journal on Imaging Science, (2009).
- A. Ahmed, B. Recht, J. Romberg, *Blind Deconvolution using Convex Programming*, IEEE Trans. on Info. Theory, (2013).

Proximal Gradient Method

$$f(x) = \frac{1}{2}\|\mathcal{A}x - b\|_2^2, \quad g(x) = \lambda\|x\|_1.$$

We can use a variant of simple gradient descent to solve the generic problem

$$\min_x f(x) + g(x).$$

Proximal gradient method:

$$x^+ = \text{prox}_{tg}\left(x - t\nabla f(x)\right), \text{ step size } t > 0.$$

Proximity function:

$$\text{prox}_g(x) := \arg\min_u \left(g(u) + \frac{1}{2}\|u - x\|_2^2 \right).$$

Proximal Gradient Method

$$\mathrm{prox}_g(x) := \arg\min_u \left(g(u) + \frac{1}{2}\|u - x\|_2^2 \right).$$

For $g(x) = \lambda\|x\|_1$, proximity operator is “shrinkage”:

$$\left\{\mathrm{prox}_g(x)\right\}_i = \begin{cases} x_i - \lambda & x_i \geq \lambda \\ 0 & |x_i| < \lambda \\ x_i + \lambda & x_i \leq -\lambda \end{cases}.$$

Proximal gradient step minimizes $g(u)$ plus quadratic local model of $f(u)$ about x:

$$\begin{aligned} x^+ &= \mathrm{prox}_{tg}\left(x - t\nabla f(x)\right) \\ &= \arg\min_u \left(g(u) + f(x) + \langle \nabla f(x), u - x\rangle + \frac{1}{2t}\|u - x\|_2^2 \right). \end{aligned}$$

Proximal Gradient and FISTA

Proximal gradient:

- Choose $x^{(0)}$.
- For $k = 1, 2, \dots$

$$x^{(k)} = \mathrm{prox}_{t_k g}\left(x^{(k-1)} - t_k \nabla f(x^{(k-1)})\right), \text{ with step size } t_k.$$

FISTA (fast iterative shrinkage-thresholding algorithm):

- Choose $x^{(0)} = x^{(-1)}$.
- For $k = 1, 2, \dots$

$$y = x^{(k-1)} + \frac{k-2}{k+1}\left(x^{(k-1)} - x^{(k-2)}\right)$$

$$x^{(k)} = \mathrm{prox}_{t_k g}\left(y - t_k \nabla f(y)\right), \text{ with step size } t_k.$$

Adjoint of Wavelet Operator

We could use $\mathcal{W}^* \approx \mathcal{W}^\dagger$, but it turns out we can find the adjoint exactly!

- Related to $\mathcal{W}$ are frame vectors ϕ_n (the wavelet basis vectors), which define a frame operator $\Phi = \mathcal{W}^\dagger$:

$$\Phi f[n] = \langle f, \phi_n \rangle.$$

- We can define the dual frame vectors $\tilde{\phi}_n = (\Phi^* \Phi)^{-1} \phi_n$.
- Define the dual frame operator via

$$\tilde{\Phi} f[n] = \langle f, \tilde{\phi}_n \rangle.$$

- Digging around in frame theory a bit, we find

$$\Phi^* = \tilde{\Phi}^\dagger \implies \mathcal{W}^* = \tilde{\mathcal{W}}^\dagger$$

Analysis formulation

Synthesis formulation

$$\min_x \frac{1}{2}\|\mathcal{R}\mathcal{W}x - b\|_2^2 + \lambda\|x\|_1$$

x contains coefficients.

Analysis formulation

$$\min_y \frac{1}{2}\|\mathcal{R}y - b\|_2^2 + \lambda\|\mathcal{W}^\dagger y\|_1$$

y is an image.

In the analysis formulation, we need $\text{prox}_g(y)$ with $g(y) = \lambda\|\mathcal{W}^\dagger y\|_1$ instead of just $\lambda\|x\|_1$.

We know how to compute the prox function if $\mathcal{W}^\dagger(\mathcal{W}^\dagger)^* = \nu I$. This is okay for orthogonal wavelets, but not for biorthogonal wavelets. But it's "close" in practice.

P. Combettes, J.-C. Pesquet, *A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery*, IEEE Journal of Selected Topics in Signal Processing, (2007).

Image Deblurring Problem

200 iterations of FISTA, $\mathcal{W}^\dagger$ is a 3-stage CDF 9/7 wavelet transform, $\lambda = 2 \times 10^{-5}$

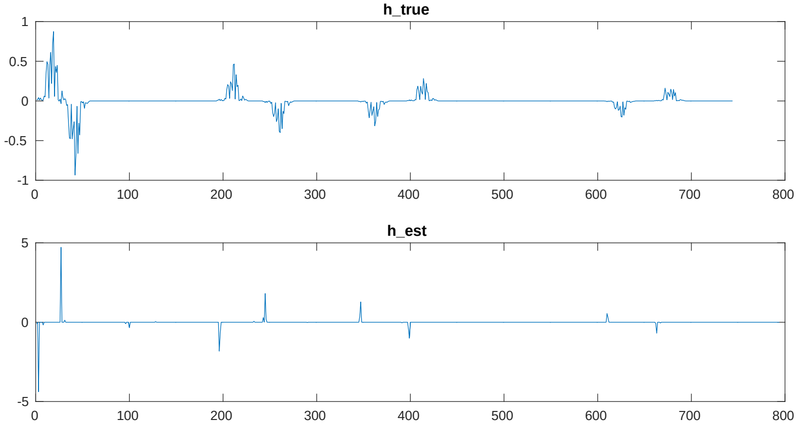
Original image:

Using $\mathcal{W}^* = \tilde{\mathcal{W}}^\dagger_{\text{zpd}}(\mathcal{E}^\dagger)^*$:

$$\frac{\|\mathcal{W}x - y\|_2}{\|y\|_2} = 7.24 \times 10^{-2}$$

Example estimation

Still a work in progress!



Example estimation

Still a work in progress!