Solving Regularized Inverse Problems with ADMM

Adapted from Gordon Wetzstein's notes by David Lindell for CSC2529 at the University of Toronto

This document serves as a supplement to the material discussed in the lectures. The document is not meant to be a comprehensive review of inverse problems, compressive sensing, or the alternating direction method of multipliers (ADMM). It is supposed to be an intuitive introduction to the basic mathematical concepts of compressive image reconstruction for the specific inverse problem of the single pixel camera using the ADMM. More information can be found in the paper by Wakin et al. [2006], which is representative for an entire class of research papers published throughout the last decade, and that by Boyd et al. [Boyd et al. 2011] on ADMM.

This set of notes uses the single-pixel camera as an interesting, yet challenging inverse problem in computational imaging. However, the same solver we derive here is directly applicable to most other inverse problems with linear image formation models. So if you want to use it for your MRI, CT, or really any other inverse problem, you can do that by following the same steps outlined here and simply use the appropriate image formation model, i.e., a different matrix $\mathbf{A}$, than that of the single-pixel camera.

1 Image Formation

Given a vectorized image $\mathbf{x} \in \mathbb{R}^N$ and a measurement matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$, the vectorized measurements $\mathbf{b} \in \mathbb{R}^M$ are formed as

$$\mathbf{b} = \mathbf{A}\mathbf{x} + \boldsymbol{\eta}. \tag{1}$$

Here, $\boldsymbol{\eta}$ is an additive, signal-independent noise term. For the single pixel camera and other compressive imaging applications, the number of measurements is usually smaller than the number of unknowns $M < N$, which makes the linear system under-determined. There are infinitely many solutions that result in the correct measurements, so which one should we pick? We have to impose a prior on the unknown image $\mathbf{x}$ that will help determine which of the infinitely many feasible solutions is more desirable than others.

The reconstruction problem is

$$\underset{\{\mathbf{x}\}}{\text{minimize}} \;\; \frac{1}{2}\left\|\mathbf{A}\mathbf{x} - \mathbf{b}\right\|_2^2 + \lambda \Psi\left(\mathbf{x}\right), \tag{2}$$

where $\Psi\left(\mathbf{x}\right)$ is the prior and λ is its relative weight compared to the data fidelity term. A popular choice for the prior is the ℓ_2-norm on the image $\|\mathbf{x}\|_2^2$. Similar to the normal equations for the least-squares solution ($\widetilde{\mathbf{x}}_{\text{ls}} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}$), which is often used to solve over-determined inverse problems, a least-norm solution is often used for under-determined problems by computing the solution to the problem minimize $\|\mathbf{x}\|_2 \quad \text{s.t.}\, \mathbf{b} = \mathbf{A}\mathbf{x}$ which is $\widetilde{\mathbf{x}}_{\text{ln}} = \mathbf{A}^T\left(\mathbf{A}\mathbf{A}^T\right)^{-1}\mathbf{b}$. Although the least-norm solution has many important applications, it may not be the solution we are looking for in many cases. Figure 1 (column 1) shows simulations of the single pixel camera for binary noise patterns in $\mathbf{A}$ with Gaussian sensor noise for a varying compression factor N/M. These solutions are computed with Python's implementation of the conjugate gradient method (via the function `scipy.sparse.linalg.cg` with matrix-free operations).

As derived in the course notes on "Image Deconvolution with the Alternating-Direction Method of Multipliers", we can re-write Eq. 2 by splitting the data fidelity term and the regularizer using an additional slack variable $\mathbf{z}$ as

$$\begin{aligned} &\underset{\{x\}}{\text{minimize}} \;\; \underbrace{\frac{1}{2}\left\|\mathbf{A}\mathbf{x} - \mathbf{b}\right\|_2^2}_{f(x)} + \underbrace{\lambda\Psi\left(\mathbf{z}\right)}_{g(z)} \\ &\text{subject to } \mathbf{D}\mathbf{x} - \mathbf{z} = 0, \end{aligned} \tag{3}$$

which we will consider the objective function of choice for the remainder of this set of notes as it lends itself well to the ADMM method.

2 Regularized Image Reconstruction with ADMM

ADMM is an iterative optimization approach that splits the objective function and solves it in an alternating manner using proximal operators for the data fidelity term and the regularization term [Boyd et al. 2011]. ADMM reformulates the objective using the Augmented Lagrangian of Equation 3 (see ADMM course notes):

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{D}\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z}\|_2^2, \tag{4}$$

where $\mathbf{y}$ is the Lagrange multiplier. As discussed in more detail in Chapter 3.1 of Boyd et al. [2011], the scaled form of the Augmented Lagrangian,

$$L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = f(\mathbf{x}) + g(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2, \tag{5}$$

will be more convenient for us, where $\mathbf{u} = (1/\rho)\mathbf{y}$ is the scaled Lagrange multiplier. Using this formulation, the following iterative updates rules can be derived:

while not converged:

$$\mathbf{x} \leftarrow \mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{v}) = \underset{\{\mathbf{x}\}}{\arg\min}\, L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \underset{\{\mathbf{x}\}}{\arg\min}\, \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{v}\|_2^2, \quad \mathbf{v} = \mathbf{z} - \mathbf{u}, \tag{6}$$

$$\mathbf{z} \leftarrow \mathbf{prox}_{\Psi,\rho}(\mathbf{v}) = \underset{\{\mathbf{z}\}}{\arg\min}\, L_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \underset{\{\mathbf{z}\}}{\arg\min}\, \lambda\Psi(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{v} - \mathbf{z}\|_2^2, \quad \mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{u}, \tag{7}$$

$$\mathbf{u} \leftarrow \mathbf{u} + \mathbf{D}\mathbf{x} - \mathbf{z}$$

The $\mathbf{x}$ and $\mathbf{z}$-updates are performed via the proximal operators $\mathbf{prox}_{\cdot,\rho}$. Since we can re-use the proximal operators of the $\mathbf{z}$-update from the other notes, i.e., the proximal operators of the TV-norm or general (learned) denoising priors, we will not derive those again here. Unlike the deconvolution problem, however, our matrix $\mathbf{A}$ is not a circulant Toeplitz matrix, so there is no closed-formed solution for the $\mathbf{x}$-update in this more general image formation model. The $\mathbf{u}$-update is trivial as it only involves a sum of vectors.

Note that $\mathbf{y}, \mathbf{u} \in \mathbb{R}^{2N}$, i.e., they have twice the number of elements as $\mathbf{x}$ when we work with the TV prior because $\mathbf{D} = [\mathbf{D}_x^T\ \mathbf{D}_y^T]^T \in \mathbb{R}^{2N \times N}$. When working with general denoising priors, $\mathbf{D} = \mathbf{I}$, so $\mathbf{y}, \mathbf{u} \in \mathbb{R}^N$ have the same number of elements as $\mathbf{x}$.

2.1 Implementation of $\mathbf{x}$-Update

For the $\mathbf{x}$-update, we need to derive the proximal operator $\mathbf{prox}_{\|\cdot\|_2,\rho}$, which is the following quadratic problem

$$\mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{v}) = \underset{\{\mathbf{x}\}}{\arg\min}\, \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{v}\|_2^2, \quad \mathbf{v} = \mathbf{z} - \mathbf{u}. \tag{8}$$

We follow the same procedure as in the notes on image deconvolution to derive the normal equations, giving the solution to this proximal operator as

$$\mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{z}) = \left(\underbrace{\mathbf{A}^T\mathbf{A} + \rho\mathbf{D}^T\mathbf{D}}_{\widetilde{\mathbf{A}}}\right)^{-1}\left(\underbrace{\mathbf{A}^T\mathbf{b} + \rho\mathbf{D}^T\mathbf{v}}_{\widetilde{b}}\right). \tag{9}$$

Unfortunately, it is not easily possible to derive a closed-form solution for the inverse of $\widetilde{\mathbf{A}}$ as it is for the deconvolution problem. Hence, we will use an iterative solver to compute this proximal operator by solving $\widetilde{\mathbf{A}}\mathbf{x} = \widetilde{\mathbf{b}}$. Since $\widetilde{\mathbf{A}}$ is symmetric and positive semi-definite, the conjugate gradient (CG) method is an adequate choice. CG is implemented in Python's SciPy package via the function `scipy.sparse.linalg.cg`. This function also supports matrix-free operations—one simply supplies a function handle that computes $\widetilde{\mathbf{A}}\mathbf{x}$ for a given vector $\mathbf{x}$ without forming the matrix. Due to symmetry of the matrix ($\widetilde{\mathbf{A}}^T = \widetilde{\mathbf{A}}$) we do not have to specify the adjoint operation. Remember that $\mathbf{D} = [\mathbf{D}_x^T\ \mathbf{D}_y^T]^T \in \mathbb{R}^{2N \times N}$ is the finite differences operator when we work with the TV prior and $\mathbf{D} = \mathbf{I} \in \mathbb{R}^{N \times N}$ when we work with general denoising priors.

2.2 Implementation of z-Update

Recall from the deconvolution course notes that the proximal operator for the TV prior is the element-wise soft thresholding operator

$$\mathbf{prox}_{\|\cdot\|_1,\rho}(\mathbf{v}) = \underset{\{\mathbf{z}\}}{\arg\min}\ \lambda\|\mathbf{z}\|_1 + \frac{\rho}{2}\|\mathbf{v}-\mathbf{z}\|_2^2 = \mathcal{S}_{\lambda/\rho}(\mathbf{v}), \tag{10}$$

where we set $\mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{u}$.

Similar to the deconvolution problem, we can work with any Gaussian denoiser $\mathcal{D}: \mathbb{R}^N \to \mathbb{R}^N$ as a regularizer. In this case, $\mathbf{D} = \mathbf{I}$ can be ignored, and we derive the proximal operator for the $\mathbf{z}$-update as

$$\mathbf{prox}_{\mathcal{D},\rho}(\mathbf{v}) = \mathcal{D}\left(\mathbf{v}, \sigma^2 = \frac{\lambda}{\rho}\right), \tag{11}$$

where $\mathbf{v} = \mathbf{x} + \mathbf{u}$. Here, the denoiser could be an implementation of the non-local means or BM3D algorithms or a learned denoiser, such as DnCNN.

2.3 Pseudo Code

Algorithms 1 and 2 outline pseudo code for solving general linear inverse problems with ADMM using TV and general denoising priors, respectively.

Algorithm 1 ADMM for solving inverse problems with TV prior

initialize ρ and λ
$\mathbf{x} = zeros(W, H)$;
$\mathbf{z} = zeros(W, H, 2)$;
$\mathbf{u} = zeros(W, H, 2)$;
for $k = 1$ **to** max_iters **do**
  $\mathbf{x} = \mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{z}-\mathbf{u}) = \text{cg\_solve}\left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{D}^T\mathbf{D}, \mathbf{A}^T\mathbf{b} + \rho\mathbf{D}^T(\mathbf{z}-\mathbf{u})\right)$
  $\mathbf{z} = \mathbf{prox}_{\|\cdot\|_1,\rho}(\mathbf{D}\mathbf{x}+\mathbf{u}) = \mathcal{S}_{\lambda/\rho}(\mathbf{D}\mathbf{x}+\mathbf{u})$
  $\mathbf{u} = \mathbf{u} + \mathbf{D}\mathbf{x} - \mathbf{z}$
end for

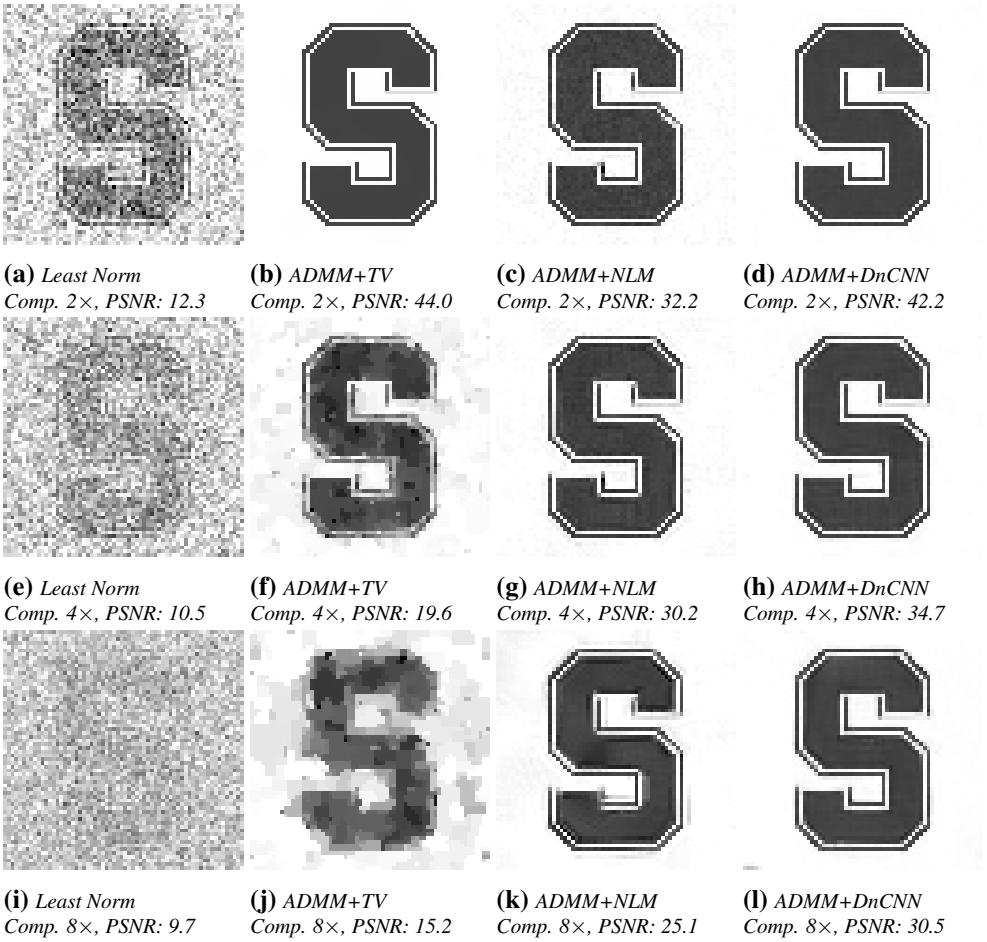
Algorithm 2 ADMM for solving inverse problems with denoising prior

initialize ρ and λ
$\mathbf{x} = zeros(W, H)$;
$\mathbf{z} = zeros(W, H)$;
$\mathbf{u} = zeros(W, H)$;
for $k = 1$ **to** max_iters **do**
  $\mathbf{x} = \mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{v}) = \text{cg\_solve}\left(\mathbf{A}^T\mathbf{A} + \rho\mathbf{I}, \mathbf{A}^T\mathbf{b} + \rho(\mathbf{z}-\mathbf{u})\right)$
  $\mathbf{prox}_{\mathcal{D},\rho}(\mathbf{x}+\mathbf{u}) = \mathcal{D}\left(\mathbf{x}+\mathbf{u}, \sigma^2 = \frac{\lambda}{\rho}\right)$
  $\mathbf{u} = \mathbf{u} + \mathbf{x} - \mathbf{z}$
end for

For additional information on implementation, please see the example code provided on the ADMM website `http://stanford.edu/~boyd/papers/admm/`.

2.4 Experiments and Results

Figure 1 shows the results of solving the compressive imaging problem with ADMM and different priors. ADMM with priors significantly outperforms the least norm approach. For low compression ratios, TV and the DnCNN prior perform the best. For higher compression ratios N/M (where the number of observations M is low compared to the number of unknowns N), NLM improves over TV, but DnCNN gives by far the best reconstruction.



(a) *Least Norm*
Comp. 2×, PSNR: 12.3

(b) *ADMM+TV*
Comp. 2×, PSNR: 44.0

(c) *ADMM+NLM*
Comp. 2×, PSNR: 32.2

(d) *ADMM+DnCNN*
Comp. 2×, PSNR: 42.2

(e) *Least Norm*
Comp. 4×, PSNR: 10.5

(f) *ADMM+TV*
Comp. 4×, PSNR: 19.6

(g) *ADMM+NLM*
Comp. 4×, PSNR: 30.2

(h) *ADMM+DnCNN*
Comp. 4×, PSNR: 34.7

(i) *Least Norm*
Comp. 8×, PSNR: 9.7

(j) *ADMM+TV*
Comp. 8×, PSNR: 15.2

(k) *ADMM+NLM*
Comp. 8×, PSNR: 25.1

(l) *ADMM+DnCNN*
Comp. 8×, PSNR: 30.5

Figure 1: *Results for single pixel imaging. Each row shows a comparison of several different solvers for an increasing compression ratio (2×, 4×, 8×). The first column shows the results of the least-norm solution, which typically fails because it does not include any priors. ADMM with either a TV or a DnCNN prior does a good job for low compression ratios, but TV begins to fail when then number of measurements becomes much smaller than the number of unknowns (rows 2 and 3). NLM and the DnCNN prior achieve the best results for high compression ratios (row 5), with DnCNN giving by far the best performance. The hyperparameters are* $\lambda = 1.0$ *and* $\rho = 16.0$ *for TV,* $\lambda = 2.1$ *and* $\rho = 4.0$ *for NLM, and* $\lambda = 0.05$ *and* $\rho = 5.2$ *for DnCNN. For all experiments with the DnCNN prior, we used the pre-trained DnCNN model for noise level 25 provided on this github repository:* `https://github.com/cszn/KAIR`.

References

BOYD, S., PARIKH, N., CHU, E., PELEATO, B., AND ECKSTEIN, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning 3*, 1, 1–122.

WAKIN, M., LASKA, J., DUARTE, M., BARON, D., SARVOTHAM, S., TAKHAR, D., KELLY, K., AND BARANIUK, R. 2006. An architecture for compressive imaging. In *Proc. International Conference on Image Processing (ICIP)*.