EE 367 / CS 448I Computational Imaging 2022
Notes: Solving Regularized Inverse Problems with ADMM
Lecture 11

Gordon Wetzstein
gordon.wetzstein@stanford.edu

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. 2001] 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

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

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 Half-quadratic Splitting Method", 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 both the HQS and ADMM methods.

2 Regularized Image Reconstruction with HQS

The HQS method included the constraints of Eq. 3 in the objective as a penalty term:

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

which resulted in the following iterative update rules

while not converged:

$$\mathbf{x} \leftarrow \mathbf{prox}_{f,\rho}\left(\mathbf{z}\right) = \underset{\{\mathbf{x}\}}{\arg\min}\, L_\rho^{(\text{HQS})}\left(\mathbf{x},\mathbf{z}\right) = \underset{\{\mathbf{x}\}}{\arg\min}\, f\left(\mathbf{x}\right) + \frac{\rho}{2}\left\|\mathbf{D}\mathbf{x}-\mathbf{z}\right\|_2^2 \tag{5}$$

$$\mathbf{z} \leftarrow \mathbf{prox}_{g,\rho}\left(\mathbf{D}\mathbf{x}\right) = \underset{\{\mathbf{z}\}}{\arg\min}\, L_\rho^{(\text{HQS})}\left(\mathbf{x},\mathbf{z}\right) = \underset{\{\mathbf{z}\}}{\arg\min}\, g\left(\mathbf{z}\right) + \frac{\rho}{2}\left\|\mathbf{D}\mathbf{x}-\mathbf{z}\right\|_2^2 \tag{6}$$

using efficient proximal operators $\mathbf{prox}_{\cdot,\rho}$ for both $\mathbf{x}$ and $\mathbf{z}$-updates. 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, so 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.

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 a quadratic program:

$$\mathbf{prox}_{\|\cdot\|_2,\rho}\left(\mathbf{v}\right) = \underset{\{\mathbf{x}\}}{\arg\min}\, \frac{1}{2}\left\|\mathbf{A}\mathbf{x}-\mathbf{b}\right\|_2^2 + \frac{\rho}{2}\left\|\mathbf{D}\mathbf{x}-\mathbf{z}\right\|_2^2 \tag{7}$$

We follow the same derivations as in the notes on image deconvolution to derive the normal equations, which allow us to derive this proximal operator as

$$\mathbf{prox}_{\|\cdot\|_2,\rho}\left(\mathbf{z}\right) = \left(\underbrace{\mathbf{A}^T\mathbf{A}+\rho\mathbf{D}^T\mathbf{D}}_{\widetilde{\mathbf{A}}}\right)^{-1}\left(\underbrace{\mathbf{A}^T\mathbf{b}+\rho\mathbf{D}^T\mathbf{z}}_{\widetilde{b}}\right). \tag{8}$$

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 Experiments

As seen in Figure 1 (columns 2, 3), the HQS approach does a good job recovering the image for a low compression factor N/M, i.e., when we have close to as many observations M as we have unknowns N. Remember that the deconvolution problem had as many observations as unknowns ($M = N$). It turns out that HQS works well in some cases, but the penalty term is a rather weak link between the data fidelity and prior terms and the overall approach does not converge well for more severely under-determined problems, such as the single-pixel camera considered here. We need a more robust solver! ADMM is exactly that, even though it is only slightly different from HQS.

3 Regularized Image Reconstruction with ADMM

Similar to HQS, 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. 2001]. Instead of re-formulating the objective (Eq. 3) using a simple penalty term (Eq. 4), ADMM re-formulates it in a slightly different way using the Augmented Lagrangian of Equation 3:

$$L_\rho^{(\text{ADMM})}(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{D}\mathbf{x} - \mathbf{z}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z}\|_2^2, \tag{9}$$

where $\mathbf{y}$ is the Lagrange multiplier. As discussed in more detail in Chapter 3.1 of Boyd et al. [2001], the scaled form of the Augmented Lagrangian,

$$L_\rho^{(\text{ADMM})}(\mathbf{x}, \mathbf{z}, \mathbf{u}) = f(\mathbf{x}) + g(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2, \tag{10}$$

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:

$$\mathbf{x} \leftarrow \mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{v}) = \underset{\{\mathbf{x}\}}{\arg\min}\, L_\rho^{(\text{ADMM})}(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \underset{\{\mathbf{x}\}}{\arg\min}\, \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2, \tag{11}$$

$$\mathbf{z} \leftarrow \mathbf{prox}_{\Psi,\rho}(\mathbf{v}) = \underset{\{\mathbf{z}\}}{\arg\min}\, L_\rho^{(\text{ADMM})}(\mathbf{x}, \mathbf{z}, \mathbf{u}) = \underset{\{\mathbf{z}\}}{\arg\min}\, \lambda\Psi(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{D}\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2 \tag{12}$$

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

Similar to HQS, the $\mathbf{x}$ and $\mathbf{z}$-updates are performed via the proximal operators $\mathbf{prox}_{\cdot,\rho}$. 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 $\mathbf{x}$.

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

The $\mathbf{x}$-update of ADMM is almost the same as that of HQS. The only difference is that we need to account for $\mathbf{u}$ as

$$\mathbf{prox}_{\|\cdot\|_2,\rho}(\mathbf{z} - \mathbf{u}) = \left(\underbrace{\mathbf{A}^T\mathbf{A} + \rho\mathbf{D}^T\mathbf{D}}_{\widetilde{\mathbf{A}}}\right)^{-1}\left(\underbrace{\mathbf{A}^T\mathbf{b} + \rho\mathbf{D}^T(\mathbf{z} - \mathbf{u})}_{\widetilde{b}}\right). \tag{13}$$

3.2 Implementation of $\mathbf{z}$-Update

ADMM uses the same proximal operators as HQS, so we can directly use the definitions derived in the deconvolution notes. Remember, the proximal operator for the TV prior is the element-wise soft thresholding operator

$$\mathbf{prox}_{\|\cdot\|_1,\rho}(\mathbf{v}) = \underset{\{\mathbf{z}\}}{\arg\min}\ \lambda\|\mathbf{z}\|_1 + \frac{\rho}{2}\|\mathbf{v}-\mathbf{z}\|_2^2 = \mathcal{S}_{\lambda/\rho}(\mathbf{v}). \tag{14}$$

The only difference between these proximal operators used with HQS and ADMM is that for HQS, $\mathbf{v} = \mathbf{D}\mathbf{x}$, and for ADMM we also have to take the variable $\mathbf{u}$ into account as $\mathbf{v} = \mathbf{D}\mathbf{x} + \mathbf{u}$.

Similar to the deconvolution problem, we can work with any Gaussian denoiser $\mathcal{D} : \mathbb{R}^N \rightarrow \mathbb{R}^N$ 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

$$\mathbf{prox}_{\mathcal{D},\rho}(\mathbf{v}) = \mathcal{D}\left(\mathbf{v}, \sigma^2 = \frac{\lambda}{\rho}\right), \tag{15}$$

where $\mathbf{v} = \mathbf{x} + \mathbf{u}$ for the ADMM method, as opposed to $\mathbf{v} = \mathbf{x}$ for HQS. As with HQS, the denoiser could use an implementation of the non-local means or BM3D algorithms or a learned denoiser, such as DnCNN.

3.3 Pseudo Code

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

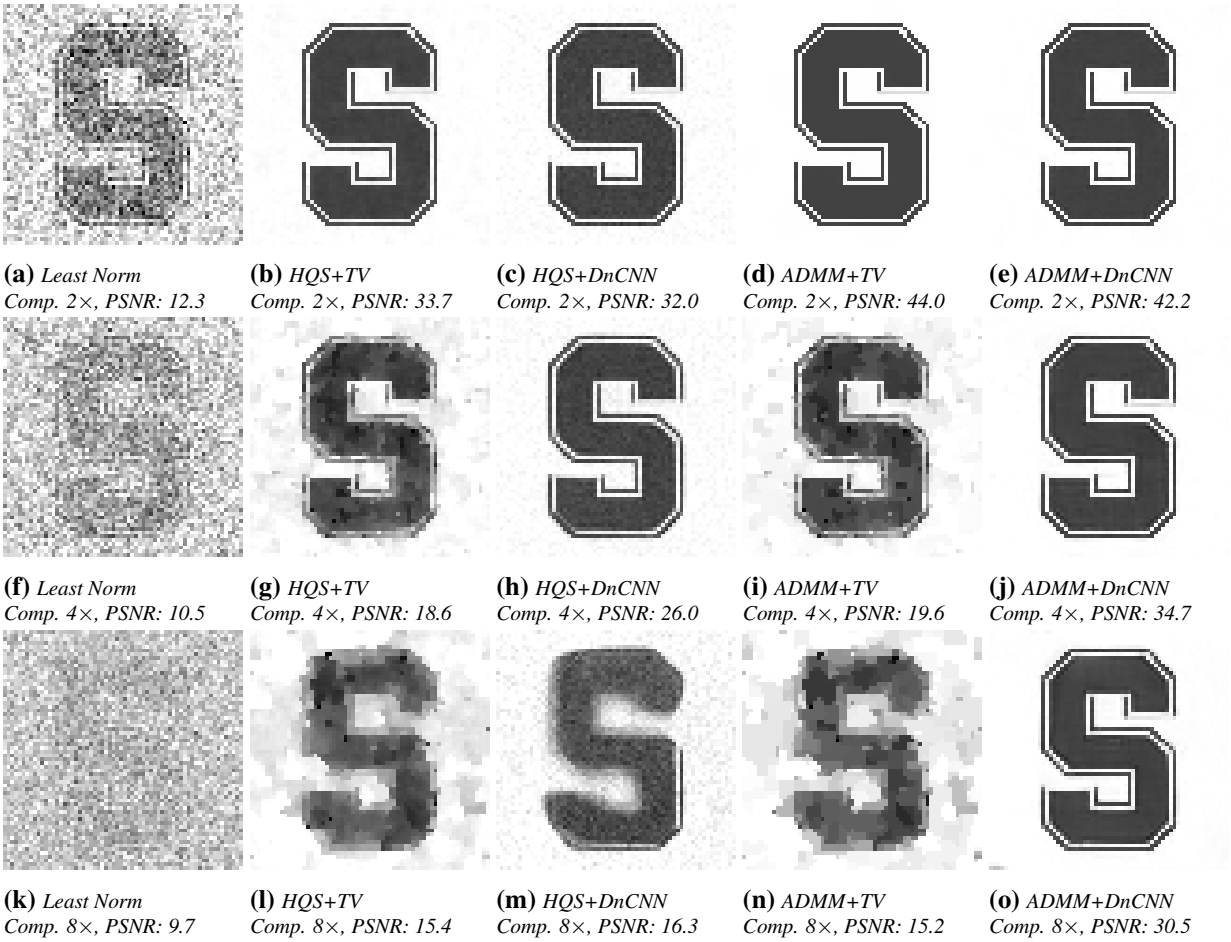
Algorithm 1 ADMM for solving inverse problems with TV prior

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/`.



(a) *Least Norm* *Comp. 2×, PSNR: 12.3*
(b) *HQS+TV* *Comp. 2×, PSNR: 33.7*
(c) *HQS+DnCNN* *Comp. 2×, PSNR: 32.0*
(d) *ADMM+TV* *Comp. 2×, PSNR: 44.0*
(e) *ADMM+DnCNN* *Comp. 2×, PSNR: 42.2*

(f) *Least Norm* *Comp. 4×, PSNR: 10.5*
(g) *HQS+TV* *Comp. 4×, PSNR: 18.6*
(h) *HQS+DnCNN* *Comp. 4×, PSNR: 26.0*
(i) *ADMM+TV* *Comp. 4×, PSNR: 19.6*
(j) *ADMM+DnCNN* *Comp. 4×, PSNR: 34.7*

(k) *Least Norm* *Comp. 8×, PSNR: 9.7*
(l) *HQS+TV* *Comp. 8×, PSNR: 15.4*
(m) *HQS+DnCNN* *Comp. 8×, PSNR: 16.3*
(n) *ADMM+TV* *Comp. 8×, PSNR: 15.2*
(o) *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. HQS with either a TV or a DnCNN prior does a good job for low compression ratios, but fails when then number of measurements becomes much smaller than the number of unknowns (rows 2 and 3). The ADMM solver achieves the best results, especially when used with the DnCNN prior (row 5). The hyperparameters for HQS and ADMM with the TV prior are* $\lambda = 1.0$ *and* $\rho = 16.0$ *and for HQS and ADMM with the DnCNN prior* $\lambda = 0.05$ *and* $\rho = 5.2$*. 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`.

3.4 Experiments and Results

Figure 1 shows that the ADMM solver is typically the most robust approach to regularized image reconstruction, especially for higher compression factors N/M when the number of observations M is low compared to the number of unknowns N. ADMM combined with a pre-trained DnCNN prior works by far the best for this problem.

References

BOYD, S., PARIKH, N., CHU, E., PELEATO, B., AND ECKSTEIN, J. 2001. 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 BARA-

NIUK, R. 2006. An architecture for compressive imaging. In *Proc. International Conference on Image Processing (ICIP)*.