Alternating direction primal dual methods

We will now focus on a class of algorithms that work by fixing the dual variables and updating the primal variables $\boldsymbol{x}$, then fixing the primals and updating the dual variables $\boldsymbol{\lambda}, \boldsymbol{\nu}$. An excellent source for this material is [BPC+10]. In fact, what follows here is basically a summary of the first 12 pages of that paper.

We have seen that when we have strong duality (which we will assume throughout), the optimal value of the primal program is equal to the optimal value of the dual program. That is, if $\boldsymbol{x}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star$ are primal/dual optimal points,

$$f(\boldsymbol{x}^\star) = d(\boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star) = \inf_{\boldsymbol{x}\in\mathbb{R}^N} \; L(\boldsymbol{x}, \boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star),$$

where L is the Lagrangian

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \sum_{m=1}^{M} \lambda_m g_m(\boldsymbol{x}) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}).$$

If $L(\boldsymbol{x}, \boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star)$ has only one minimizer,[1] then we can recover the primal optimal solution $\boldsymbol{x}^\star$ from the dual-optimal solution $\boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star$ by solving the *unconstrained* program

$$\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x}\in\mathbb{R}^N} \; L(\boldsymbol{x}, \boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star).$$

"Alternating" methods search for a saddle point of the Lagrangian by fixing the dual variables $\boldsymbol{\lambda}_k, \boldsymbol{\nu}_k$, minimizing $L(\boldsymbol{x}, \boldsymbol{\lambda}_k, \boldsymbol{\nu}_k)$ with respect to $\boldsymbol{x}$, then updating the Lagrange multipliers.

To start, we will base our discussion on **equality constrained** problems. Incorporating inequality constraints will be natural after we have developed things a bit.

[1] Which is the case when f is strictly convex, and in many other situations.

Dual ascent

We want to solve

$$\underset{\boldsymbol{x}\in\mathbb{R}^N}{\text{minimize}}\ f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}.$$

We will assume that the domain of f is all of $\mathbb{R}^N$; again, things are easily modified if this is any open set. The Lagrangian is

$$L(\boldsymbol{x},\boldsymbol{\nu}) = f(\boldsymbol{x}) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}),$$

and the dual function is

$$\begin{aligned} d(\boldsymbol{\nu}) &= \inf_{\boldsymbol{x}}\ L(\boldsymbol{x},\boldsymbol{\nu}) \\ &= \inf_{\boldsymbol{x}}\ f(\boldsymbol{x}) + \boldsymbol{\nu}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{\nu}^{\mathrm{T}}\boldsymbol{b} \\ &= -\sup_{\boldsymbol{x}}\left((-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\nu})^{\mathrm{T}}\boldsymbol{x} - f(\boldsymbol{x})\right) - \boldsymbol{\nu}^{\mathrm{T}}\boldsymbol{b} \\ &= -f^*(-\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\nu}) - \boldsymbol{\nu}^{\mathrm{T}}\boldsymbol{b}, \end{aligned}$$

and the dual problem is

$$\underset{\boldsymbol{\nu}\in\mathbb{R}^N}{\text{maximize}}\ d(\boldsymbol{\nu}).$$

Consider for a moment the problem of maximizing the dual. A reasonable thing to do would be some kind of gradient ascent:[2]

$$\boldsymbol{\nu}_{k+1} = \boldsymbol{\nu}_k + \alpha_k \nabla d(\boldsymbol{\nu}_k),$$

where α_k is some appropriate step size. The gradient of d (with respect to $\boldsymbol{\nu}$) is

$$\nabla d(\boldsymbol{\nu}) = \nabla_{\boldsymbol{\nu}}\left(\inf_{\boldsymbol{x}}\ f(\boldsymbol{x}) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})\right).$$

[2] "Ascent" instead of "descent" because $d(\boldsymbol{\nu})$ is concave instead of convex.

Taking $\boldsymbol{x}^+ = \arg\min_{\boldsymbol{x}} \; f(\boldsymbol{x}) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$, we have

$$\begin{aligned}\nabla d(\boldsymbol{\nu}) &= \nabla_{\boldsymbol{\nu}} \left(f(\boldsymbol{x}^+) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x}^+ - \boldsymbol{b}) \right) \\ &= \boldsymbol{A}\boldsymbol{x}^+ - \boldsymbol{b}.\end{aligned}$$

This leads naturally to:

The **dual ascent** algorithm consists of the iteration

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \; L(\boldsymbol{x}, \boldsymbol{\nu}_k) \\ \boldsymbol{\nu}_{k+1} &= \boldsymbol{\nu}_k + \alpha_k(\boldsymbol{A}\boldsymbol{x}_{k+1} - \boldsymbol{b})\end{aligned}$$

that is repeated until some convergence criteria is met.

This algorithm "works" under certain assumptions on f (that translate to different assumptions on the dual d). In particular, we need $L(\boldsymbol{x}, \boldsymbol{\nu})$ to be bounded for every $\boldsymbol{\nu}$, otherwise the primal update $\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}} \; L(\boldsymbol{x}, \boldsymbol{\nu}_k)$ can fail.

That the Lagrangian is bounded below for every choice of $\boldsymbol{\nu}$ is far from a given. Looking at

$$L(\boldsymbol{x}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}),$$

we can see that if f increases slowly (sublinearly) in any direction even partially aligned with the row space of $\boldsymbol{A}$, then we can find a sequence of $\boldsymbol{x}$ that drive $L(\boldsymbol{x}, \boldsymbol{\nu}) \to -\infty$ for certain $\boldsymbol{\nu}$.

Of course, this algorithm is nicest when we can solve the unconstrained primal update problem efficiently.

The Method of Multipliers and Augmented Lagrangians

The method of multipliers (MoM) is the same idea as dual ascent, but we smooth out (augment) the Lagrangian to make the primal update more robust.

It should be clear that

$$\underset{\boldsymbol{x}\in\mathbb{R}^N}{\text{minimize}}\ f(\boldsymbol{x}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},$$

and

$$\underset{\boldsymbol{x}\in\mathbb{R}^N}{\text{minimize}}\ f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

have exactly the same set of solutions for all $\rho \geq 0$.

The Lagrangian for the second program is

$$L_\rho(\boldsymbol{x}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}).$$

This is called the **augmented Lagrangian** of the original problem.

Adding the quadratic term is nice – it makes (under mild conditions on f with respect to $\boldsymbol{A}$) the primal update minimization well-posed (i.e., makes the dual differentiable).

Notice that the Lagrange multipliers $\boldsymbol{\nu}$ appear in exactly the same way in the augmented Lagrangian as they do in the regular Lagrangian, so the dual update does not change.

The resulting algorithm is called the **method of multipliers**; we iterate

$$\begin{aligned}\boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \ L_\rho(\boldsymbol{x}, \boldsymbol{\nu}_k) \\ \boldsymbol{\nu}_{k+1} &= \boldsymbol{\nu}_k + \rho(\boldsymbol{A}\boldsymbol{x}_{k+1} - \boldsymbol{b})\end{aligned}$$

until some convergence criteria is met.

As a bonus, we now have a principled way of selecting the step size for the dual update – just use ρ. To see why this makes sense, recall the KKT conditions for $\boldsymbol{x}^\star$ and $\boldsymbol{\nu}^\star$ to be a solution:

$$\boldsymbol{A}\boldsymbol{x}^\star = \boldsymbol{b}, \quad \nabla f(\boldsymbol{x}^\star) + \boldsymbol{A}^{\mathrm{T}}\boldsymbol{\nu}^\star = \boldsymbol{0}.$$

With ρ as the step size, we have

$$\begin{aligned}\boldsymbol{0} &= \nabla_{\boldsymbol{x}} L_\rho(\boldsymbol{x}_{k+1}, \boldsymbol{\nu}_k), \quad \text{(since } \boldsymbol{x}_{k+1} \text{ is a minimizer)}, \\ &= \nabla f(\boldsymbol{x}_{k+1}) + \boldsymbol{A}^{\mathrm{T}}\left(\boldsymbol{\nu}_k + \rho(\boldsymbol{A}\boldsymbol{x}_{k+1} - \boldsymbol{b})\right) \\ &= \nabla f(\boldsymbol{x}_{k+1}) + \boldsymbol{A}^{\mathrm{T}}\boldsymbol{\nu}_{k+1}.\end{aligned}$$

So the dual update maintains the second optimality condition at every step.

The MoM has much better convergence properties than dual ascent. The algorithm we look at next, the alternating direction method of multipliers (ADMM), will build on this idea in a way that makes it applicable to functions that have a nonsmooth component and can be easily modified to incorporate certain kinds of inequality constraints.

Alternating direction method of multipliers (ADMM)

ADMM **splits** the optimization variables into two parts, $\boldsymbol{x}$ and $\boldsymbol{z}$, and solves programs of the form

$$\underset{\boldsymbol{x},\boldsymbol{z}}{\text{minimize}} \;\; f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} = \boldsymbol{c}.$$

The basic idea is to rotate through 3 steps:

1. Minimize the (augmented) Lagrangian over $\boldsymbol{x}$ with $\boldsymbol{z}$ and the Lagrange multipliers $\boldsymbol{\nu}$ fixed.
2. Minimize the (augmented) Lagrangian over $\boldsymbol{z}$ with $\boldsymbol{x}$ and $\boldsymbol{\nu}$ fixed.
3. Update the Lagrange multipliers using gradient ascent as before.

If the splitting is done in a careful manner, it can happen that each of the subproblems above can be easily computed. We can also handle general convex constraints through projection by taking g to be an indicator function (more on this later).

To make the three steps above more explicit: the augmented Lagrangian is

$$L_\rho(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\nu}) = f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c}\|_2^2,$$

and the general ADMM iteration is

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \; L_\rho(\boldsymbol{x}, \boldsymbol{z}_k, \boldsymbol{\nu}_k) \\
\boldsymbol{z}_{k+1} &= \arg\min_{\boldsymbol{z}} \; L_\rho(\boldsymbol{x}_{k+1}, \boldsymbol{z}, \boldsymbol{\nu}_k) \\
\boldsymbol{\nu}_{k+1} &= \boldsymbol{\nu}_k + \rho(\boldsymbol{A}\boldsymbol{x}_{k+1} + \boldsymbol{B}\boldsymbol{z}_{k+1} - \boldsymbol{c}).
\end{aligned}$$

The only real difference between ADMM and MoM is the we are splitting the primal minimization into two parts instead of optimizing over $(\boldsymbol{x}, \boldsymbol{z})$ jointly.

Scaled form.

We can write the ADMM iterations in a more convenient form by substituting

$$\boldsymbol{\mu} = \frac{1}{\rho}\boldsymbol{\nu}.$$

By "completing the square" we have that

$$\boldsymbol{\nu}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}\boldsymbol{z}-\boldsymbol{c})+\frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}\boldsymbol{z}-\boldsymbol{c}\|_2^2 = \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x}+\boldsymbol{B}\boldsymbol{z}-\boldsymbol{c}+\boldsymbol{\mu}\|_2^2-\frac{\rho}{2}\|\boldsymbol{\mu}\|_2^2,$$

and so we can write:

ADMM:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{z}_k - \boldsymbol{c} + \boldsymbol{\mu}_k\|_2^2 \right) \\
\boldsymbol{z}_{k+1} &= \arg\min_{\boldsymbol{z}} \left(g(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{A}\boldsymbol{x}_{k+1} + \boldsymbol{B}\boldsymbol{z} - \boldsymbol{c} + \boldsymbol{\mu}_k\|_2^2 \right) \\
\boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \boldsymbol{A}\boldsymbol{x}_{k+1} + \boldsymbol{B}\boldsymbol{z}_{k+1} - \boldsymbol{c}
\end{aligned}$$

Example: The LASSO

Recall the LASSO:

$$\underset{\boldsymbol{x}}{\text{minimize}}\ \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\tau\|\boldsymbol{x}\|_1.$$

Taking

$$f(\boldsymbol{x})=\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 \quad \text{and} \quad g(\boldsymbol{z})=\tau\|\boldsymbol{z}\|_1,$$

we can rewrite this in ADMM form as

$$\underset{\boldsymbol{x},\boldsymbol{z}}{\text{minimize}}\ f(\boldsymbol{x})+g(\boldsymbol{z}) \quad \text{subject to} \quad \boldsymbol{x}-\boldsymbol{z}=\boldsymbol{0}.$$

The $\boldsymbol{x}$ update is

$$\boldsymbol{x}_{k+1}=\arg\min_{\boldsymbol{x}}\left(\frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2+\frac{\rho}{2}\|\boldsymbol{x}-\boldsymbol{z}_k+\boldsymbol{\mu}_k\|_2^2\right).$$

With both $\boldsymbol{z}_k$ and $\boldsymbol{\mu}_k$ fixed, this is a regularized least-squares problem and is equivalent to:

$$\min_{\boldsymbol{x}}\ \left\|\begin{bmatrix}\boldsymbol{A}\\ \sqrt{\rho}\,\mathbf{I}\end{bmatrix}\boldsymbol{x}-\begin{bmatrix}\boldsymbol{b}\\ \sqrt{\rho}(\boldsymbol{z}_k-\boldsymbol{\mu}_k)\end{bmatrix}\right\|_2^2.$$

This problem has a closed-form solution:

$$\begin{aligned}\boldsymbol{x}_{k+1}&=\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\rho\mathbf{I}\right)^{-1}\begin{bmatrix}\boldsymbol{A}^{\mathrm{T}} & \sqrt{\rho}\,\mathbf{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{b}\\ \sqrt{\rho}(\boldsymbol{z}_k-\boldsymbol{\mu}_k)\end{bmatrix}\\ &=\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}+\rho\mathbf{I}\right)^{-1}\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}+\rho(\boldsymbol{z}_k-\boldsymbol{\mu}_k)\right)\end{aligned}$$

The $\boldsymbol{z}$ update problem is:

$$\underset{\boldsymbol{z}}{\text{minimize}} \;\; \tau\|\boldsymbol{z}\|_1 + \frac{\rho}{2}\|\boldsymbol{z} - \boldsymbol{x}_{k+1} - \boldsymbol{\mu}_k\|_2^2.$$

You may recognize this: it is the proximal operator for the ℓ_1-norm, which as we have seen before has closed form solution:

$$\boldsymbol{z}_{k+1} = T_{\tau/\rho}(\boldsymbol{x}_{k+1} + \boldsymbol{\mu}_k),$$

where $T_\lambda(\cdot)$ is the term-by-tern soft-thresholding operator,

$$(T_\lambda(\boldsymbol{v}))[n] = \begin{cases} v[n] - \lambda, & v[n] > \lambda, \\ 0, & |v[n]| \leq \lambda, \\ v[n] + \lambda, & v[n] < -\lambda. \end{cases}$$

To summarize:

ADMM iterations for the LASSO

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} + \rho\mathbf{I}\right)^{-1}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b} + \rho(\boldsymbol{z}_k - \boldsymbol{\mu}_k)), \\ \boldsymbol{z}_{k+1} &= T_{\tau/\rho}(\boldsymbol{x}_{k+1} + \boldsymbol{\mu}_k), \\ \boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}. \end{aligned}$$

Convergence properties

We will state one convergence result. If the following two conditions hold:

1. f and g are closed, proper, and convex (i.e., their epigraphs are nonempty closed convex sets),
2. strong duality holds,

then

- $\boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{z}_k - \boldsymbol{c} \to \boldsymbol{0}$ as $k \to \infty$. That is, the primal iterates are asymptotically feasible.
- $f(\boldsymbol{x}_k) + g(\boldsymbol{z}_k) \to f(\boldsymbol{x}^\star) + g(\boldsymbol{x}^\star)$ as $k \to \infty$. That is, the value of the objective function approaches the optimal value asymptotically.
- $\boldsymbol{\nu}_k \to \boldsymbol{\nu}^\star$ as $k \to \infty$, where $\boldsymbol{\nu}^\star$ is a dual optimal point.

Under additional assumptions, we can also have convergence to a primal optimal point, $(\boldsymbol{x}_k, \boldsymbol{z}_k) \to (\boldsymbol{x}^\star, \boldsymbol{z}^\star)$ as $k \to \infty$.

See [BPC+10, Section 3.2] for further discussion and references.

Convex constraints

Using a technique that we have seen before, we can write the general program

$$\underset{\boldsymbol{x} \in \mathcal{C}}{\text{minimize}} \;\; f(\boldsymbol{x}),$$

where $\mathcal{C}$ is a closed convex set, in ADMM form as

$$\underset{\boldsymbol{x} \in \mathbb{R}^N}{\text{minimize}} \;\; f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{subject to} \quad \boldsymbol{x} - \boldsymbol{z} = \boldsymbol{0},$$

where $g(\boldsymbol{z})$ is the indicator function for $\mathcal{C}$:

$$g(\boldsymbol{z}) = \begin{cases} 0, & \boldsymbol{z} \in \mathcal{C}, \\ \infty, & \boldsymbol{z} \notin \mathcal{C}. \end{cases}$$

Note that in this case, the $\boldsymbol{z}$ update is a closest-point-to-a-convex-set problem. For fixed $\boldsymbol{v} \in \mathbb{R}^N$:

$$\arg\min_{\boldsymbol{z}} \;\; g(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{z} - \boldsymbol{v}\|_2^2 = \arg\min_{\boldsymbol{z} \in \mathcal{C}} \;\; \|\boldsymbol{z} - \boldsymbol{v}\|_2 = \mathcal{P}_{\mathcal{C}}(\boldsymbol{v}).$$

ADMM iteration for general convex constraints:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \left(f(\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{z}_k + \boldsymbol{\mu}_k\|_2^2 \right), \\ \boldsymbol{z}_{k+1} &= \mathcal{P}_{\mathcal{C}}\left(\boldsymbol{x}_{k+1} + \boldsymbol{\mu}_k\right), \\ \boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}. \end{aligned}$$

Of course, this algorithm is most attractive when we have a fast method for computing $\mathcal{P}_{\mathcal{C}}(\cdot)$.

Example: Basis Pursuit

As we have seen before, a good proxy for finding the sparsest solution to an underdetermined system of equations $\boldsymbol{Ax} = \boldsymbol{b}$ is to solve

$$\underset{\boldsymbol{x}}{\text{minimize}} \ \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \boldsymbol{Ax} = \boldsymbol{b}.$$

To put this in ADMM form, we are solving

$$\underset{\boldsymbol{x},\boldsymbol{z}}{\text{minimize}} \ f(\boldsymbol{x}) + g(\boldsymbol{z}) \quad \text{subject to} \quad \boldsymbol{x} - \boldsymbol{z} = \mathbf{0},$$

with

$$f(\boldsymbol{x}) = \|\boldsymbol{x}\|_1, \quad \text{and} \quad g(\boldsymbol{z}) = \begin{cases} 0, & \boldsymbol{Az} = \boldsymbol{b}, \\ \infty, & \text{otherwise.} \end{cases}$$

The projection onto $\mathcal{C} = \{\boldsymbol{x} : \boldsymbol{Ax} = \boldsymbol{b}\}$ can be given in closed form using the pseudo-inverse c of $\boldsymbol{A}$ as

$$\begin{aligned} \mathcal{P}_{\mathcal{C}}(\boldsymbol{v}) &= \boldsymbol{A}^{\dagger}(\boldsymbol{b} - \boldsymbol{Av}) + \boldsymbol{v} \\ &= (\mathbf{I} - \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{AA}^{\mathrm{T}})^{-1}\boldsymbol{A})\boldsymbol{v} + \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{AA}^{\mathrm{T}})^{-1}\boldsymbol{b}, \end{aligned}$$

where the last equality comes from $\boldsymbol{A}^{\dagger} = \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{AA}^{\mathrm{T}})^{-1}$ when $\boldsymbol{A}$ has full row rank.

The updates in this case are

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \left(\|\boldsymbol{x}\|_1 + \frac{\rho}{2}\|\boldsymbol{x} - \boldsymbol{z}_k + \boldsymbol{\mu}_k\|_2^2 \right) \\ &= T_{1/\rho}(\boldsymbol{z}_k - \boldsymbol{\mu}_k) \\ \boldsymbol{z}_{k+1} &= (\mathbf{I} - \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{AA}^{\mathrm{T}})^{-1}\boldsymbol{A})(\boldsymbol{x}_{k+1} + \boldsymbol{\mu}_k) + \boldsymbol{A}^{\mathrm{T}}(\boldsymbol{AA}^{\mathrm{T}})^{-1}\boldsymbol{b} \\ \boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \boldsymbol{x}_{k+1} - \boldsymbol{z}_{k+1}. \end{aligned}$$

Example: Linear programming

Consider the general linear program

$$\underset{\boldsymbol{x}}{\text{minimize}}\ \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x} \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \quad \boldsymbol{x} \geq \boldsymbol{0},$$

where $\boldsymbol{A}$ is an $M \times N$ matrix with full row rank.[3] We can put this in ADMM form by first eliminating the equality constraints, then introducing the indicator function for the non-negativity constraint.

Let $\boldsymbol{Q}$ be an $N \times (N-M)$ matrix whose columns span $\text{Null}(\boldsymbol{A})$, and let $\boldsymbol{x}_0$ be any point such that $\boldsymbol{A}\boldsymbol{x}_0 = \boldsymbol{b}$. Then we can re-write the LP as

$$\underset{\boldsymbol{w}}{\text{minimize}}\ \boldsymbol{c}^{\mathrm{T}}(\boldsymbol{x}_0 + \boldsymbol{Q}\boldsymbol{w}) \quad \text{subject to} \quad \boldsymbol{x}_0 + \boldsymbol{Q}\boldsymbol{w} \geq \boldsymbol{0},$$

which we can write in ADMM form as

$$\underset{\boldsymbol{w}}{\text{minimize}}\ \boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}_0 + \boldsymbol{c}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{w} + g(\boldsymbol{z}) \quad \text{subject to} \quad \boldsymbol{Q}\boldsymbol{w} - \boldsymbol{z} = -\boldsymbol{x}_0,$$

where

$$g(\boldsymbol{z}) = \begin{cases} 0, & \boldsymbol{z} \geq \boldsymbol{0}, \\ \infty, & \text{otherwise.} \end{cases}$$

(We can drop the $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{x}_0$ from the objective since it does not depend on either of the optimization variables.)

Notice that when $\boldsymbol{Q}$ has full column rank, the program

$$\underset{\boldsymbol{w}}{\text{minimize}}\ \boldsymbol{v}^{\mathrm{T}}\boldsymbol{w} + \frac{1}{2}\|\boldsymbol{Q}\boldsymbol{w} - \boldsymbol{y}\|_2^2,$$

[3]The full row rank assumption is not at all essential; I am just making it to keep things streamlined.

has the closed-form solution

$$\boldsymbol{w}^\star = (\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q})^{-1}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{y} - \boldsymbol{v}).$$

Also, the projection onto the non-negative orthant $\mathcal{C} = \{\boldsymbol{x} : \boldsymbol{x} \geq \boldsymbol{0}\}$ is

$$\mathcal{P}_{\mathcal{C}}(\boldsymbol{v}) = (\boldsymbol{v})_+, \quad \text{or} \quad (\mathcal{P}_{\mathcal{C}}(\boldsymbol{v}))[n] = \begin{cases} v[n], & v[n] \geq 0, \\ 0, & v[n] < 0. \end{cases}$$

For the general linear program, then, the ADMM iterations are

$$\begin{aligned}
\boldsymbol{w}_{k+1} &= \arg\min_{\boldsymbol{w}} \left(\frac{1}{\rho}\boldsymbol{c}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{w} + \frac{1}{2}\|\boldsymbol{Q}\boldsymbol{w} - \boldsymbol{z}_k + \boldsymbol{x}_0 + \boldsymbol{\mu}_k\|_2^2 \right) \\
&= (\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q})^{-1}\left[\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{z}_k - \boldsymbol{x}_0 - \boldsymbol{\mu}_k) - \frac{1}{\rho}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{c}\right], \\
\boldsymbol{z}_{k+1} &= P_{\mathcal{C}}(\boldsymbol{Q}\boldsymbol{w}_{k+1} + \boldsymbol{x}_0 + \boldsymbol{\mu}_k) \\
&= (\boldsymbol{Q}\boldsymbol{w}_{k+1} + \boldsymbol{x}_0 + \boldsymbol{\mu}_k)_+ \\
\boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \boldsymbol{Q}\boldsymbol{w}_{k+1} - \boldsymbol{z}_{k+1} + \boldsymbol{x}_0.
\end{aligned}$$

Notice that especially when the columns of $\boldsymbol{Q}$ are orthogonal, $\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q} = \mathbf{I}$, all of these steps are very simple.

References

[AHU58] K. J. Arrow, L. Hurwicz, and H. Uzawa. *Studies in Linear and Non-Linear Programming.* Stanford University Press, 1958.

[Bec17] A. Beck. *First-order methods in optimization.* SIAM, 2017.

[BPC+10] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

[CP11] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *J. Math. Imaging and Vision*, 40(1):120–145, 2011.