Algorithms for constrained optimization

There are as many constrained optimization algorithms as there are pages in the SIAM Journal of Optimization. But many can categorized into a small number of basic frameworks.

Eliminating equality constraints

Programs with linear equality constraints can always be written as programs without, regardless of whether there are other inequality constraints — if there are not, the new program is unconstrained. Suppose we are solving

\[
\min_{x \in \mathbb{R}^N} f_0(x) \quad \text{subject to} \quad Ax = b.
\]

Let \( x_0 \) be any feasible point, \( Ax_0 = b \). Then we can re-write the program above as a search over \( \text{Null}(A) \):

\[
\min_h f_0(x_0 + h) \quad \text{subject to} \quad h \in \text{Null}(A).
\]

Since \( \text{Null}(A) \) is a linear subspace of dimension \( K = N - \text{rank}(A) \), there is an \( N \times K \) matrix \( Q \) whose columns span \( \text{Null}(A) \) (in fact, there are many such matrices). Note that for such \( Q \), we will have \( AQ = 0 \). We can now re-write the program above as

\[
\min_{w \in \mathbb{R}^K} f_0(x_0 + Qw).
\]

Sometimes this method can be very helpful, and sometimes it does not make things much easier — it could be the case that computing \( x_0 \) and \( Q \) is as expensive as just solving the program directly.
Barrier methods

We saw a little of this already. The basic idea is to enforce the inequality constraints by introducing a penalty in the objective that blows up at the boundary of the feasible set. The constrained program

$$\min_x f_0(x) \text{ subject to } f_m(x) \leq 0, \ m = 1, \ldots, M$$

becomes

$$\min_x f_0(x) + \sum_{m=1}^{M} B(f_m(x)),$$

where $\text{dom} \ B = \mathbb{R}_-$ and $B(x) \to \infty$ as $x \to 0$ from the left. Of course, unless $B$ is the indicator $1_{x<0}$, the new program is an approximation to the original.

An interesting choice is $B(x) = -\tau \log(-x)$. This particular barrier function has the properties

- You can analyze the number of Newton iterations needed for convergence for many $f_0$ of interest (using self-concordance). This is done very nicely in Chapters 9 and 11 of [BV04].
- The solution $\mathbf{x}(\tau)^*$ of

$$\min_x \tau f_0(x) - \sum_{m=1}^{M} \log(-f_m(x))$$

can be used to generate a dual-feasible point (and hence a primal-dual gap certificate), and be related to the KKT conditions for the original program.

\[1\]We have multiplied the objective by $\tau$ to make some of what follows a little easier to express.
To appreciate the second point above, we start by taking the gradient of the functional above and setting it equal to zero. We see that

$$\tau \nabla f_0(\mathbf{x}^*(\tau)) + \sum_{m=1}^{M} -\frac{1}{f_m(\mathbf{x}^*(\tau))} \nabla f_m(\mathbf{x}^*(\tau)) = \mathbf{0}.$$ 

So if we take

$$\lambda_m^*(\tau) = -\frac{1}{\tau f_m(\mathbf{x}^*(\tau))}, \quad m = 1, \ldots, M,$$

we have $$\lambda_m^*(\tau) \geq 0$$ and

$$f_0(\mathbf{x}^*(\tau)) + \sum_{m=1}^{M} \lambda_m^*(\tau) \nabla f_m(\mathbf{x}^*(\tau)) = \mathbf{0}.$$ 

Since $$\mathbf{x}^*(\tau)$$ is primal feasible, the only KKT condition we are missing is complementary slackness — we have replaced the condition

$$\lambda_m^* f_m(\mathbf{x}^*) = 0, \quad \text{with} \quad \lambda_m^*(\tau) f_m(\mathbf{x}^*(\tau)) = -1/\tau.$$ 

So as $$\tau$$ gets large, we are producing points that meet the KKT conditions.

With this choice of $$\lambda^*(\tau)$$, we can also easily compute the dual of the original program:

$$g(\lambda^*(\tau)) = \inf_{\mathbf{x}} \left( f_0(\mathbf{x}) + \sum_{m=1}^{M} \lambda_m^*(\tau) f_m(\mathbf{x}) \right)$$

$$= f_0(\mathbf{x}^*(\tau)) + \sum_{m=1}^{M} \lambda_m^*(\tau) f_m(\mathbf{x}^*(\tau))$$

$$= f_0(\mathbf{x}^*(\tau)) - \frac{m}{\tau}.$$
So then we know that if $p^*$ is the primal optimal value of the original program,

$$f_0(x^*(\tau)) - p^* \leq f_0(x^*(\tau)) - g(\lambda^*(\tau)) \leq m/\tau.$$ 

So we know that solving the log-barrier problem gets us within $m/\tau$ of the optimal of the original primal objective.

**Primal dual interior point methods**

These are closely related to log barrier algorithms, but they take a more direct approach towards ”solving” the KKT conditions. The general idea is to treat the KKT conditions like a set of nonlinear equations, and solve them using Newton’s method.

We start with the same set of relaxed KKT conditions we used with log barrier:

$$r_{\tau}(x, \lambda) = \begin{bmatrix} \nabla f_0(x) + \sum_m \lambda_m \nabla f_m(x) \\ -\lambda_1 f_1(x) - 1/\tau \\ \vdots \\ -\lambda_m f_m(x) - 1/\tau \end{bmatrix}$$

If we find $x$ and $\lambda$ such that the $N + M$-vector $r_{\tau}(x, \lambda) = 0$, then we know we have found the same $x^*(\tau), \lambda^*(\tau)$ that solve the log barrier problem.

Primal-dual interior point methods take Newton steps to try to make $r_{\tau} = 0$, but they adjust $\tau$ at every step. The Newton step is characterized by

$$r_{\tau}(x + \delta x, \lambda + \delta \lambda) \approx r_{\tau}(x, \lambda) + J_r(x, \lambda) \begin{bmatrix} \delta x \\ \delta \lambda \end{bmatrix} = 0,$$

\(^2\)We are again only considering inequality constraints; it is straightforward to modify everything we say here to include linear equality constraints.
where $J_{r\tau}$ is the Jacobian matrix for the vector-valued function $r_{\tau}$:

$$J_{r\tau}(x, \lambda) = \begin{bmatrix}
\nabla^2 f_0(x) + \sum_m \lambda_m \nabla^2 f_m(x) & \nabla f_1(x) & \nabla f_2(x) & \cdots & \nabla f_m(x) \\
-\lambda_1 \nabla f_1(x)^T & -f_1(x) & 0 & \cdots & 0 \\
-\lambda_2 \nabla f_2(x)^T & 0 & -f_2(x) & \cdots & 0 \\
\vdots & & & \ddots & \\
-\lambda_M \nabla f_M(x)^T & 0 & 0 & \cdots & -f_M(x)
\end{bmatrix}.$$ 

The update direction is

$$\begin{bmatrix} \delta x \\ \delta \lambda \end{bmatrix} = -J_{r\tau}^{-1} r_{\tau}(x, \lambda).$$

With this direction, we can perform a line search. The parameter $\tau$ is updated at every step (getting larger) based on an estimate of the duality gap.

A key feature of this type of primal dual method is that the iterates $x^{(k)}$ and $\lambda^{(k)}$ do not have to be feasible (although they of course become feasible in the limit).

Details on this particular algorithm, along with a full convergence analysis, can be found in [BV04, Chap. 11.7].
Alternating direction primal dual methods

We will focus on a class of algorithms that work by fixing the dual variables and updating the primal variables $x$, then fixing the pri-
mals and updating the dual variables $\lambda, \nu$. An excellent source for this material is \cite{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 $\mathbf{x}^\ast, \lambda^\ast, \nu^\ast$ are primal/dual optimal points,

$$f_0(\mathbf{x}^\ast) = g(\lambda^\ast, \nu^\ast) = \inf_{\mathbf{x} \in \mathbb{R}^N} \mathcal{L}(\mathbf{x}, \lambda^\ast, \nu^\ast),$$

where $\mathcal{L}$ is the Lagrangian

$$\mathcal{L}(\mathbf{x}, \lambda, \nu) = f_0(\mathbf{x}) + \sum_{m=1}^{M} \lambda_m f_m(\mathbf{x}) + \nu^T(A \mathbf{x} - \mathbf{b}).$$

If $\mathcal{L}(\mathbf{x}, \lambda^\ast, \nu^\ast)$ has only one minimizer\footnote{Which is the case when $f_0$ is strictly convex, and in many other situations.}, then we can recover the primal optimal solution $\mathbf{x}^\ast$ from the dual-optimal solution $\lambda^\ast, \nu^\ast$ by solving the unconstrained program

$$\mathbf{x}^\ast = \arg \min_{\mathbf{x} \in \mathbb{R}^N} \mathcal{L}(\mathbf{x}, \lambda^\ast, \nu^\ast).$$

“Alternating” methods search for a saddle point of the Lagrangian by fixing the dual variables $\lambda^{(k)}, \nu^{(k)}$, minimizing $\mathcal{L}(\mathbf{x}, \lambda^{(k)}, \nu^{(k)})$ with respect to $\mathbf{x}$, 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.

**Dual ascent**

We want to solve

$$\min_x f_0(x) \text{ subject to } Ax = b.$$  

We will assume that the domain of $f_0$ is all of $\mathbb{R}^N$; again, things are easily modified if this is any open set. The Lagrangian is

$$L(x, \nu) = f_0(x) + \nu^T(Ax - b),$$

and the dual is

$$g(\nu) = \inf_x L(x, \nu)$$

$$= -f_0^*(-A^T\nu) - \langle \nu, b \rangle,$$

where $f_0^*$ is the Fenchel conjugate of $f_0$:

$$f_0^*(y) = \sup_{x \in \mathbb{R}^N} (\langle x, y \rangle - f_0(x)).$$

The dual problem is

$$\max_{\nu \in \mathbb{R}^N} g(\nu).$$

Consider for a moment the problem of maximizing the dual. A reasonable thing to do would be some kind of gradient ascent$^4$,

$$\nu^{(k+1)} = \nu^{(k)} + t_k \nabla g(\nu^{(k)}),$$

$^4$“Ascent” instead of “descent” because $g$ is concave instead of convex.
where $t_k$ is some appropriate step size. The gradient of $g$ at a point $\nu_0$ is

$$\nabla g(\nu_0) = \nabla_\nu \inf_x (f_0(x) + \langle \nu_0, A x - b \rangle),$$

and so if $x^+ = \arg\min_x (f_0(x) + \langle \nu_0, A x - b \rangle)$, then

$$\nabla_\nu g(\nu_0) = \nabla_\nu (f_0(x^+) + \langle \nu_0, A x^+ - b \rangle) = A x^+ - b.$$

This leads naturally to:

The dual ascent algorithm consists of the iteration

$$x^{(k+1)} = \arg\min_x L(x, \nu^{(k)})$$

$$\nu^{(k+1)} = \nu^{(k)} + t_k (A x^{(k+1)} - b)$$

that is repeated until some convergence criteria is met.

This algorithm “works” under certain assumptions on $f_0$ (that translate to different assumption on the dual $g$). In particular, we need $L(x, \nu)$ to be bounded for every $\nu$, otherwise the primal update $x^{(k+1)} = \arg\min_x L(x, \nu^{(k)})$ can fail.

That the Lagrangian is bounded below for every choice of $\nu$ is far from a given. For example, a program of the form

$$\min_{x_1, x_2} f_0(x_1) + \langle x_2, c \rangle \text{ subject to } A_1 x_1 + A_2 x_2 = b$$

will have Lagrangian

$$L(x_1, x_2, \nu) = f_0(x_1) + \langle x_2, c \rangle + \nu^T(A_1 x_1 + A_2 x_2 - b)$$

$$= f_0(x_1) + \nu^T(A_1 x_1 - b) + (c + A_2^T \nu)^T x_2,$$
which is unbounded below since the last term is linear in $x_2$.

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

**Dual decomposition**

Dual ascent is simple, and it is pretty much as old an idea as convex optimization itself. But it has a key feature that makes it very attractive (at least as a starting point) for modern computing platforms.

If the objective functional $f_0$ is *separable*, we can also separate (i.e. parallelize) the primal update. Suppose we can write $f_0(x)$ as a sum

$$f_0(x) = \sum_{i=1}^{B} f_i(x_i),$$

where the $x_i \in \mathbb{R}^{N_i}$ ($\sum_i N_i = N$) are a partition of $x$. We can partition the columns of $A$ in the same way,

$$A = [A_1 A_2 \cdots A_B]$$

so that

$$Ax = \sum_{i=1}^{B} A_ix_i.$$

Notice that even with this assumption, the primal optimization program itself is not separable, as the constraints $Ax = b$ tie all of the blocks together — there is still just a single right-hand side vector $b$. 
The Lagrangian in this case is also separable:

\[
L(x, \nu) = \sum_{i=1}^{B} f_i(x_i) + \nu^T \left( \sum_{i=1}^{B} A_i x_i \right) - \nu^T b
\]

\[
= \sum_{i=1}^{B} \left[ f_i(x_i) + \nu^T A_x i - \frac{1}{B} \nu^T b \right]
\]

\[
= \sum_{i=1}^{B} L_i(x_i, \nu).
\]

For fixed \( \nu^{(k)} \), the primal update

\[
x^{(k+1)} = \arg \min_x L(x, \nu^{(k)})
\]

can be separated into \( B \) independent updates:

\[
x^{(k+1)}_i = \arg \min_{x_i} L_i(x_i, \nu)
\]

You can imagine broadcasting the current dual iterate to \( B \) different processors; they each compute there piece of the primal update \( x^{(k+1)}_i \), and then the results are gathered centrally to compute the dual update. Notice also, though, that the hard part of the dual update, computing \( A x^{(k+1)} \), can also be done in a decentralized manner.
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
\[
\min_x f_0(x) \quad \text{subject to} \quad Ax = b,
\]
and
\[
\min_x f_0(x) + \frac{\rho}{2} \|Ax - b\|_2^2 \quad \text{subject to} \quad Ax = b
\]
have exactly the same set of solutions for all \( \rho \geq 0 \).

The Lagrangian for the second program is
\[
L_\rho(x, \nu) = f_0(x) + \frac{\rho}{2} \|Ax - b\|_2^2 + \nu^T(Ax - b).
\]
This is called the augmented Lagrangian of the original problem.

Adding the quadratic term is nice — it makes (under mild conditions on \( f_0 \) with respect to \( A \)) the primal update minimization well-posed (i.e. makes the dual differentiable).

Notice that the Lagrange multipliers \( \nu \) appear in exactly the same was 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

\[
    \begin{align*}
    \mathbf{x}^{(k+1)} &= \arg\min_{\mathbf{x}} L_{\rho}(\mathbf{x}, \nu^{(k)}) \\
    \nu^{(k+1)} &= \nu^{(k)} + \rho(\mathbf{A}\mathbf{x}^{(k+1)} - \mathbf{b})
    \end{align*}
\]

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 \(\rho\). To see why this makes sense, recall the KKT conditions for \(\mathbf{x}^*\) and \(\nu^*\) to be a solution:

\[
    \mathbf{A}\mathbf{x}^* = \mathbf{b}, \quad \nabla f_0(\mathbf{x}^*) + \mathbf{A}^T\nu^* = \mathbf{0}.
\]

With \(\rho\) as the step size, we have

\[
    \begin{align*}
    0 &= \nabla_x L_{\rho}(\mathbf{x}^{(k+1)}, \nu^{(k)}), \quad \text{(since } \mathbf{x}^{(k+1)} \text{ is a minimizer)}, \\
    &= \nabla f_0(\mathbf{x}^{(k+1)}) + \mathbf{A}^T\left(\nu^{(k)} + \rho(\mathbf{A}\mathbf{x}^{(k+1)} - \mathbf{b})\right) \\
    &= \nabla f_0(\mathbf{x}^{(k+1)}) + \mathbf{A}^T\nu^{(k+1)}.
    \end{align*}
\]

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

The MOM has much better convergence properties than dual ascent, but it is no longer separable. The algorithm we look at next, the alternating direction method of multipliers (ADMM), will build on this idea in such a way that we do have a type of dual decomposition for the smoothed Lagrangian. In addition, it can be easily modified to incorporate certain kinds of inequality constraints.
References
