Linear Programming:
A Classic Example of Constrained Convex Optimization

Henry Ou
University of California Los Angeles

Abstract

Linear programming is one of the simplest possible forms of constrained optimization. It is of particular interest because it has an existing solution technique in the form of the simplex method which guarantees an exact solution in exponential time. This provides both a great way to judge exactly how close an optimization technique is to the true solution and raises hopes that an optimization technique may be able to reach a solution within a small ε of the true solution in polynomial time.

1 The Problem

Linear programming problems are typically represent in the following canonical form:

$$\begin{aligned} \max c^{\mathsf{T}} x \\ \text{subject to } Ax \leq b \\ \text{and } x \geq 0 \end{aligned}$$

where x is the vector of variables to be determined, c and b are known vectors, A is a known matrix, and $\geq / \leq$ are the entry-wise inequalities. We note in particular that x can be extended to the real line by taking component-wise $x_i = x_i^* - x_i^{**}$ where $x_i^*, x_i^{**} \geq 0$.

1.1 Interpretation

The problem is a simple linear function constrained to exist within the intersection of finitely many half spaces in $\mathbb{R}^n_+$, each defined by the entry-wise linear inequalities in $Ax \leq b$. The result is a convex polytope, possibly unbounded.

2 Applications

Despite the simplicity of the problem in regards to both the objective function and the domain, the problem is broadly applicable in a variety of fields. Problems in assignment, routing, scheduling can all frequently be modelled as a linear relationship over some domain of feasible solutions (in the real world sense). The idea of restricting the problem to one on a convex polytope is likewise not particularly limiting, as it is appropriate to approximate any bounded convex domain in $\mathbb{R}^n$ by a polytope [2].

3 Properties

3.1 Coercivity

The above function is clearly not coercive, as choosing $x = -sc^{\mathsf{T}}$, $s \in \mathbb{R}_+$, we get that $\lim_{s\to\infty} c^{\mathsf{T}}(-sc) = lim_{s\to\infty} -s\|c\|^2 \to -\infty$.

3.2 Continuity

The objective function is linear, which is clearly continuous and differentiable over all of $\mathbb{R}^n$, and thus also continuous and differentiable over our domain of consideration.

$$|c^\intercal x - c^\intercal y| = |c^\intercal (x-y)| \leq \|c\|\|x-y\|$$

and

$$\|\nabla f(x) - \nabla f(y)\| = \|c-c\| = 0 \leq k\|x-y\|$$

So we have that the function is both Lipschitz continuous and smooth.

3.3 Convexity

Let us first check that the constraints result in a convex domain. $x \geq 0$ as a linear combination of positive vectors with positive coefficients is still clearly positive. Letting x and x^* be such that $Ax \leq b$ and $Ax^* \leq b$, we have that $A(px + (1-p)x^*) = A(px) + A(1-p)x^* \leq pb + (1-p)b = b$. Thus each condition results in a convex domain, and the intersection of convex sets is itself convex. As the objective function is linear, we have that $c^\intercal(tx_0 + (1-t)x_1) = tc^\intercal x_0 + (1-t)c^\intercal x_1$ for all x_0 and x_1 in our domain. By the equality, we have that the objective function is both convex and concave.

3.4 Bounded Hessian

We notice that $D^2(c^\intercal x) = 0$ so that $D^2(c^\intercal x) \preceq \lambda I$ for all $\lambda \in \mathbb{R}_+$.

3.5 Minimizers

We note that the derivative of the objective function is c which is non-zero at any given point x, so we know that any minimizer must exist at the boundary of the domain. It is clear that if the given domain is unbounded or empty, that it is possible for no minimizer to exist. Otherwise, we have a minimizer from the continuity and compactness of the problem. Thus we can limit our search for the minimizer to the boundary of the polytope.

We can further reduce the search space to simply the vertices of the polytope, as we can take the projection of the subsets of the hyperplanes represent the faces of the polytope onto the line spanned by the gradient vector. It is clear that the extremal points of the projected interval must be either the projection of the whole face, an edge, or and vertex. As all three possibilities contain the vertices, at least one optimal solution must be a vertex.

3.6 Duality

We shall formulate the Lagrangian dual problem for linear programming. The Lagrangian is as follows

$$\min_{x\geq 0}\max_{\lambda\geq 0} c^\intercal x + \lambda^\intercal(Ax-b)$$

We notice that the above is equivalent to

$$\min_{x\geq 0}\max_{\lambda\geq 0}(c^\intercal + \lambda^\intercal A)x - \lambda^\intercal b$$

$$\min_{x\geq 0}\max_{\lambda\geq 0} -b^\intercal\lambda + x^\intercal(A^\intercal\lambda + c)$$

$$\max_{x\geq 0}\max_{\lambda\geq 0} b^\intercal\lambda - x^\intercal(A^\intercal\lambda + c)$$

$$\max_{\lambda\geq 0}\max_{x\geq 0} b^\intercal\lambda - x^\intercal(A^\intercal\lambda + c)$$

$$\max_{\lambda\geq 0}\max_{x\leq 0} b^\intercal\lambda + x^\intercal(A^\intercal\lambda + c)$$

which is the lagrange multiplier of the following dual problem:

$$\begin{aligned} \max b^\intercal\lambda \\ \text{subject to } A^\intercal\lambda \geq c \\ \text{and } \lambda \geq 0 \end{aligned}$$

It is clear that by repeating this procedure for the dual problem, we reobtain a problem of the canonical form. Letting p^* and d^* be the optimal values of the primal and dual problems respectively. Then by the Weak Duality Theorem, we have that

$$p^* \leq d^* \leq p^*$$

and as such $p^* = d^*$. The problem is strongly dual.

4 Simplex

4.1 Idea

From the above result on the existence of minimizers at the vertices of the polytope, we can try searching the vertices one at a time.

4.2 Implementation

Implementation details are omitted as this paper is focused on more traditional optimization techniques.

4.3 Convergence

On a machine with infinite precision, we note that the process must result in the exact value in finite time because the process exhaustively searches through the vertices of the polytope, of which there are a finite number. The only concern is that cycling could possibly occur during the move from vertex to vertex the algorithm flip flops between these two vertices. There exist existing implementations of the pivot from vertex to vertex that avoids cycling [1].

4.4 Limitations

There exist strategies to remove from consideration specific vertices as to reduce the number of searches necessary. However, the fact remains that for a dimension n polytope, there are $O(2^n)$ vertices. Moreover, to the best of our knowledge, given any method of pivoting from one vertex to another, it is possible to construct a worst case polytope for which the simplex algorithm has to visit a non-polynomial many of these vertices. The Klee-Minty cube is a classic example that causes most simplex method variants to visit every vertex [5].

5 Proximal Gradient Descent

5.1 Idea

Given an algorithm that has guaranteed exact solutions but bad time complexity, we would like to perform the standard trick for a problem of this type. We relax the condition that requires the solution to be exact (real world machines are finite precision anyways!) and instead iteratively try to obtain a "close enough" solution.

Given that the problem is convex, a first idea might be proximal gradient descent.

5.2 Implementation

Let C be the set where $x \geq 0$ and $Ax \leq b$. We can consider optimizing the function $f(x) = c^\intercal x + \mathbb{1}_C(x)$, where

$$\mathbb{1}_C(x) = \begin{cases} 0 & \text{if } x \in C \\ +\infty & \text{otherwise} \end{cases}$$

As our objective function must take finite values on any bounded domain, the optimal value must reside within the constrained domain and the objective function and $f(x)$ agree on the constrained set. Thus the two problems are equivalent. Then taking the proximal operator of $\mathbb{1}_C(x)$, we get

$$\begin{aligned} &\arg\min_x \mathbb{1}_C(x) + \frac{\lambda}{2}\|x - v_n\|^2 \\ =&\arg\min_x \begin{cases} \frac{\lambda}{2}\|x - v_n\|^2 & \text{if } x \in C \\ +\infty & \text{otherwise} \end{cases} \\ =&argmin_{x\in C}\frac{\lambda}{2}\|x - v_n\|^2 \end{aligned}$$

which is simply the projection onto C. The method reduces to projected gradient descent.

Algorithm 1: Projected Gradient Descent

Result: Solution to the linear programming problem

$\hat{x}_0 \in \mathbb{R}^n_+$, $L > 0$

$x_0 = \arg\min y \in C\|x - y\|_2^2$

while *Stopping condition not reached* **do**

  $\hat{x}_{k+1} = x_k - \frac{1}{L}c$

  $x_{k+1} = \arg\min y \in C\|\hat{x}_k - y\|_2^2$

end

5.3 Convergence

The Hessian of the objective function is bounded by λI for all $\lambda \geq 0$, so any choice of step size is possible. While it may be tempting to improve the convergence rate of the method by picking immensely large choices of step size, we need to take care to avoid avoid situations where finite precision can lead to numerical instability. We know that if the objective function is Lipschitz smooth and convex and the domain C is convex, we have that projected gradient descent with step-size $\frac{1}{L}$ satisfies

$$f(x_k) - f(x^*) \leq \frac{L}{2k} \|x^* - x_0\|_2^2$$

and thus we have the method converges within ε of the optimum with $O(\frac{L}{\varepsilon})$ steps [7]. Notice that by taking the step size to infinity, this method becomes a one step method.

5.4 Starting Condition

As this method assumes an orthogonal projection into the set is easy, it suffices to pick any point on the interior of the domain, as for a bounded domain, this provides an upper bound on $\|x^* - x_0\|_2^2$.

5.5 Stopping Condition

If we know that the polytope is bounded by some ball of radius R, by the above convergence result, it suffices to choose k sufficiently large such that $\frac{2LR^2}{k} \leq \varepsilon$. This gives us an upper bound on the number of iterations necessary. It is generally reasonable to assume some known finite bound on R is known in real world problems.

Sometimes, with a good initial guess convergence occurs much faster than the upper bound k requires. In these cases, a common technique is to pick some μ such that $\|x_{k+1} - x_k\| \leq \mu$. This does not have the same convergence guarantees, but as the objective function is well-behaved, it serves as an appropriate substitute.

5.6 Limitations

Although the convex polytope seems like a simple geometry, projection unto the domain $Ax \leq b$ and $x \geq 0$ is a very difficult task to perform in actuality. As a consequence, we can only apply this technique to polytopes with specific geometries, like a hypercube. This greatly reduces the applicability of the method to linear programming problems. In addition, these simple polytopes also have relevant pivoting rules that generally result in linear performance, defeating the primary purpose of moving to an iterative method.

5.7 Other Notes

We argue briefly that once projected gradient descent has stepped our point outside the constraints, the projection will heretofore remain on the boundary. This is because any projection from outside the set onto the set results in a boundary point. Furthermore by the convexity of the constrained set, there must be no more points in the direction of the gradient direction. As such, projected gradient descent is a method that walks along the faces of the polytope, instead of just the vertices.

6 Primal Dual Hybrid Gradient

6.1 Idea

The standard projected gradient descent is less useful than we would hope as we have difficulty projecting unto the set C where $Ax \leq b$ and $x \geq 0$. However, it is easy to project unto the sets $Ax = b$ and $x \geq 0$ individually. The constraints $Ax \leq B$ and $x \geq 0$ can be converted into the above form through the introduction of slack variabls s though $Ax + Is = b$ and $x, s \geq 0$. We would like to in some sense decouple the projection onto these two seperate sets into the optimization method, which the primal-dual class of algorithms provides.

6.2 Implementation

Primal-dual hybrid gradient works on a function of the form

$$\min_{x \in X} \max_{y \in Y} f(x) + y^{\mathsf{T}} Ax - g(y)$$

where f represents the primal problem and g represents the dual problem. It is clear from the above argument on the duality of the linear programming problem that the Lagrangian exactly satisfies this as

$$\min_{x\geq 0}\max_{y\geq 0} c^{\intercal}x + y^{\intercal}(Ax-b)$$
$$= \min_{x\geq 0}\max_{y\geq 0} c^{\intercal}x + y^{\intercal}Ax - y^{\intercal}b$$

where $f(x) = c^{\intercal}x$ and $g(y) = y^{\intercal}b$, the primal and dual problems respectively.

Algorithm 2: Primal-Dual Hybrid Gradient

Result: Solution to the linear programming problem

$x_0 \in \mathbb{R}^n_+$, $y_0 \in \mathbb{R}^m_+$, $\sigma > 0$, $\tau > 0$

while *Stopping condition not reached* **do**

$\hat{x}_{k+1} = x_k - \tau A^{\intercal}y$

$x_{k+1} = \arg\min_{x\geq 0} c^{\intercal}x + \frac{1}{2\tau}\|x - \hat{x}_{k+1}\|^2$

$\bar{x}_{k+1} = x_{k+1} + \Theta(x_{k+1} - x_k)$

$\hat{y}_{k+1} = y_k + \sigma A\bar{x}_{k+1}$

$y_{k+1} = \arg\min_{y\geq 0} b^{\intercal}y + \frac{1}{2\sigma}\|y - \hat{y}_{k+1}\|^2$

end

In particular, $c^{\intercal}x + \frac{1}{2\tau}\|x - \hat{x}_{k+1}\|^2$ is a convex function with derivative $c + \frac{1}{\tau}(x - \hat{x}_{k+1})$, so that the optimal x is $\hat{x}_{k+1} - \tau c$ without domain restrictions. As the above function is clearly coercive (it is quadratic with a positive coefficient on the squared term), we have that $x_{k+1} = \max(0, \hat{x}_{k+1} - \tau c)$. A similar argument applies for y_{k+1}.

6.3 Convergence

Letting $\Theta = 1$ and $\tau\sigma \leq \frac{1}{L^2}$, where $L = \|A\|$, the standard matrix norm, with a finite dimensional polytope, we have the following convergence results:

$x_k \to x*$ and $y_k \to y*$ where (x^*, y^*) is a saddle point of our Lagrangian function.

Let $x_N = (\sum_1^N x_n)/N$ and $y_N = (\sum_1^N y_n)/N$. For any bounded $B_1 \times B_2 \in X \times Y$, we have that

$$G_{B_1\times B_2}(x_N, y_N) \leq \frac{1}{N}\Big(\sup_{(x,y)\in B_1\times B_2} \frac{\|x-x_0\|^2}{2\tau} + \frac{\|y-y_0\|^2}{2\sigma}\Big)$$

where $G_{B_1\times B_2}(x,y) = \max_{y^*\in B_2} f(x) + y^{*\intercal}Ax - g(y^*) - \min_{x^*\in B_1} f(x^*) + y^{\intercal}Ax^* - g(y)$. the partial primal-dual gap. This value is zero only if (x,y) is a saddle point, and therefore an optimal solution [3].

6.4 Starting Point

Similar to proximal gradient descent, having a starting point inside the domain gives an upper bound on $\|x - x_0\|^2$ and $\|y - y_0\|^2$ if the domain is bounded. As $Ax \leq b$ and $x \geq 0$ are both convex sets, the Projection Onto Convex Sets method gives a linear time convergence algorithm onto the interior of the intersection of the two sets (not necessarily orthogonal) [4].

6.5 Stopping Condition

Using the same argument as in proximal gradient descent, it suffices to choose an R such that the domain is bounded by a ball of radius R, and then pick an N sufficiently large so that the partial primal-dual gap satisfies a specific tolerance.

Likewise, terminating the iteration when improvement gets sufficiently small is also a common technique for any convergent method, but without the same guarantees the above has. Note that trying to actually calculate the partial primal-dual gap is again a linear programming problem.

6.6 Limitations

Observing the below results, it is clear that the PDHG method does not perform well on general linear programming problems. However the primal dual idea of solving linear programs has been shown to work quite well for some for some extremely large structured programs arising in image processing. [6]

7 Miscellaneous

7.1 Other Ideas

The current state of the art for linear programming is generally considered to be the simplex methods and interior point methods. Interior point methods expands on the idea of traversing the vertices/faces of the polytope to the interior of the polytope (which the optimal solution cannot lie). This is done through the use of barrier functions in which the value rises rapidly towards $+\infty$ as a point approaches the boundary, guaranteeing that the iterate remains in the interior of the polytope. A parameter μ is iteratively decreased allowing the next iterate to get closer to the boundary before suffering the rapid increase in objective value.

7.2 Integer Linear Programming

The integer linear programming problems requires that $x \in \mathbb{N}^n$ instead of $x \geq 0$ and cannot be solved using any of the above methods in general.

8 Results

The code is written in matlab using varying levels of optimization and do not represent the full potential of each method. The results are only provided for analyzing trends.

8.1 Hypercube

The following results are based on a linear programming problem constrained to lie within a hypercube as to make projection in the proximal gradient descent case easy. The true solution is easy to obtain, by picking the extremal value in each dimension as to minimize the objective.

The problem is so easy that the preprocessor for Matlab's linprog routine found the solution without iteration.

Observe the nearly perfectly linear decrease in iteration count as step size if increased. This supports the convergence results, and supports the idea that the method could be a one step method if only the projection onto the domain were easy.

Method	Size	Iterations	Time (s)
Proximal Gradient Descent	100	174	0.016243
	200	26	0.012325
	400	1187	0.034683
	800	451	0.030349
Primal Dual Hybrid Gradient	100	168	0.052629
	200	28	0.059517
	400	1151	3.294508
	800	410	4.977218

Table 1: Proximal Gradient Descent with step-size 1. PDHG with $\tau, \sigma = 1$. $\varepsilon = 1e^{-8}$

Step-size	Iterations	Time (s)
0.1	24997	0.710039
1	2500	0.220781
10	250	0.074505
100	25	0.025276

Table 2: Proximal Gradient Descent with 1000 dimensions. $\varepsilon = 1e^{-8}$.

We observe that proximal gradient descent does indeed decrease linearly with increases in step size. This would result in an amazing one step convergence method for linear programming if only the projection were easy.

8.2 General Convex Polytope

We assume that the simplex result is the true solution when measuring the convergence of PDHG, which is a reasonable assumption given that the method is close to machine precision exact. Note that the simplex method used here operates on the dual problem. From the results, PDHG require far more iterations that either simplex and interior point methods, and correspondingly orders of magnitude more time.

Method	Size	Iterations	Time (s)
Simplex	5	1	0.226895
	20	3	0.039679
	80	16	0.033798
PDHG	5	593	0.013561
	20	3479	0.095417
	80	400810	20.087422
Interior Point	5	6	0.008945
	20	8	0.045574
	80	13	0.226315

Table 3: PDHG with $\tau = \sigma = \frac{1}{||A||}$ and $\varepsilon = 1e^{-6}$.

Availability

All code used can be found on https://github.com/zeroblackalpha/Math273A.

References

[1] New finite pivoting rules for the simplex method. *Math. Oper. Res.*, 2(2):103–107, May 1977.

[2] E. M. Bronstein. Approximation of convex sets by polytopes. *Journal of Mathematical Sciences*, 153(6):727–762, Sep 2008.

[3] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. working paper or preprint, Jun 2010.

[4] L.G. Gubin, B.T. Polyak, and E.V. Raik. The method of projections for finding the common point of convex sets. *USSR Computational Mathematics and Mathematical Physics*, 7(6):1 – 24, 1967.

[5] V. Klee and G.J. Minty. How good is the simplex algorithm? *Inequalities*, III:159–175.

[6] Thomas Pock and Antonin Chambolle. Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In *Proceedings of ICCV 2011*. ., 2011.

[7] Michael Saunders. Notes on first-order methods for minimizing smooth functions. Lecture Notes.