Two-Phase Simplex Algorithm and Duality

CS 149 Staff

October 20, 2007

1 Finding Initial Basic Feasible Solution

So far we have assumed that we have one basic feasible solution and we want to optimize its objective value. For many problems setting all the non-slack variables to zero yields a feasible solution, but that does not work if the right-hand side vector has negative components. The iterative improvement technique that worked so well for optimizing the objective also works well to achieve feasibility. There are two approaches, both of which work by relaxing the problem so that finding a basic feasible solution is trivial, and then working to restore the original problem while maintaining feasibility. The easier approach for humans is to repeatedly pick one of the violated constraints, and making pivots until that constraint is no longer violated (ignoring the other violated constraints). Another approach, which is useful for the primal dual algorithm, is to add new slack variables that indicate how much each constraint is violated, and then minimize the sum of these artificial variables.

For the first approach, the first step is to do Gaussian elimination to obtain a basic solution. (If the problem was in canonical form, the slack variables would already be a basis and this step is trivial.) Note which variables in the initial basic solution are negative. Temporarily ignore these variables and the corresponding constraints, updating the constraints in the same manner as the objective row but not considering them when determining which variable should leave the basis. The parts of the tableau that aren't being ignored are a feasible basis for a less constrained problem. The algorithm proceeds by selecting one of the violated constraints and using it as the objective function, attempting to minimize the contribution of the other variables to the left hand side so that the negative variable can be made positive. Note that to guarantee termination one must keep with a particular violated constraints until it is satisfied rather than switching from one constraint to another before the first constraint is satisfied. Also, if the problem of satisfying a constraint is unbounded, let the objective constraint enter the basis. The slides give two examples of using this method.

An alternate approach considers all of the violated constraints at once. In this approach, start with some (not necessarily basic) solution and negate every row that has a negative right hand side. This fixes the negative right hand side problem, but eliminates any basis we started with. Introduce an artificial variable for each row indicating how much that constraint is violated and add that variable to the left hand side. These artificial variables form the initial basis. For example consider

finding a solution to $-x_1 - x_2 - x_3 = -2$ and $2x_1 + x_2 + 3x_3 - 2x_4 = 6$. The resulting tableau is:

a_1	a_2	x_1	x_2	x_3	x_4	
1	0	1	1	1	0	2
0	1	2	1	3	-2	6

Note that a point is a feasible solution to the original problem if and only if all the artificial variables are zero. Therefore, replace the original objective with minimizing the sum of the artificial variables.

a_1	a_2	x_1	x_2	x_3	x_4	
1	1	0	0	0	0	0
1	0	1	1	1	0	2
0	1	2	1	3	-2	6

The reduced costs over the basic variables must be zero, so subtract all of the rows from the new objective:

a_1	a_2	x_1	x_2	x_3	x_4	
0	0	-3	-2	-4	2	-8
1	0	1	1	1	0	2
0	1	2	1	3	-2	6

Run the simplex algorithm on this modified problem. For the running example the result is:

a_1	a_2	x_1	x_2	x_3	x_4	
4	0	1	2	0	2	0
1	0	1	1	1	0	2
-3	1	$\boxed{-1}$	$\boxed{-2}$	0	$\boxed{-2}$	0

If the optimal objective is positive, the original problem is infeasible. If the optimal objective is zero (as in this case), we have a feasible solution to the original problem. Unfortunately artificial variables might be in the final basis but degenerately equal to zero, so summarily removing the extra variables would leave us without a basis. For example, the example final tableau has objective value zero but a_2 is in the basis. To fix this make arbitrary pivots in the offending row (boxed in the above example). Pivoting around a negative entry is not a problem because the corresponding right-hand side is zero.

Finally, eliminate the artificial columns from the tableau, replace the artificial objective with the real objective, do row operations to restore zeros over the basic variables, and proceed with the second phase of the simplex algorithm.

2 Runtime

We now have an algorithm that can solve any linear program. The worst-case run time, however, is bounded by the number of bases, which is not polynomial. In the

1980s two algorithms that solve linear programs in polynomial time were discovered: the ellipsoid method and the interior point method.[1] One of the great mysteries in combinatorial optimization in the later half of the 20th century was why the simplex algorithm works so well in practice despite its poor worst-case performance. Recently a few studies have shed some light on this issue. One can show that for some models of random linear programs the simplex algorithm terminates, on average, in polynomial time. Apparently from the perspective of the simplex algorithm real-world instances have more in common with random instances then worst-case ones[2]. A recent paper gives a randomized algorithm that is inspired by but rather different from the simplex algorithm that has polynomial expected runtime. See http://theory.csail.mit.edu/∼kelner/PDFs/KelnerSpielmanSimplex.pdf for details.

3 Duality

Duality concerns the generation of lower bounds on solutions to a linear programming (traditionally the primal is a minimization problem and the dual is a maximization problem). Suppose a linear program in canonical form has objective function minimize $2x + 3y$ and one of the constraints is $2x + 3y \geq 7$. Clearly in this case 7 is a lower bound on the objective value. Now suppose the constraint is changed to $2x + y \geq 7$. Since $y \geq 0$, this constraint is strictly stronger, so the same lower bound should hold. To show this, add the valid inequality $2y \geq 0$ to $2x + y \geq 7$ yielding $2x+3y \geq 7$. As a final example, consider a problem with the same objective function and constraints $2x + y \geq 7$ and $x + 2.5y \geq 5$. Considering the first constraints alone yields a lower bounds of seven as before. We can multiply the second constraint by $6/5$ yielding a lower bound of 6. If one instead adds half the first inequality to the second, one obtains $2x + 3y \geq 8.5$, better than either of the other bounds.

One natural question is what are the best bounds obtainable from this technique. To answer this question a little formality is required. Consider a canonical form linear program minimize $c^T x$ subject to $Ax \geq b$ and $x \geq 0$. What we did was find a linear combination of the constraints $Ax \geq b$ such that each of the coefficients in front of variables x_i is at most c_i. Let $\pi \geq 0$ be the weights of each constraint. It turns out that taking a linear combination of the constraints is equivalent to pre-multiplying A and b by π^T, so for any feasible point x we have $\pi^T Ax \geq \pi^T b$. Suppose π satisfies $\pi^T A \leq c^T$. Then $\pi^T b \leq \pi^T Ax \leq c^T x$. This is called weak duality.

Suppose we wish to find the best possible lower bound of this sort. This problem can be formulated as a linear program, maximize $\pi^T b$ subject to $\pi \geq 0$ and $\pi^T A \leq$

[1] The ellipsoid algorithm actually works on a more general class of problems: minimization of a linear function over a convex set, as long as there is a polynomial time algorithm (called a separation oracle) that takes a point as input and if the point is not in the set, produces a hyper plane separating that point from the set. This can even be extended to minimization of the convex function over a convex set. The interior point method is also useful for nonlinear programs.

[2] In contrast, simple-minded quicksort with the pivot always being the first element of the input exhibits worst-case behavior with a very realistic input: the input list being already sorted. Whether worst or average case are more representative of real-world instances depends on the problem.

c^T. This is called the dual of the original problem. It turns out that the dual of the dual is the original, primal problem. To show this, rewrite the dual problem in canonical form as minimize $-b^T\pi$ subject to $\pi \geq 0$ and $-A^T\pi \geq -c$. Taking the dual of this yields maximize $-c^T y$ subject to $y \geq 0$ and $-y^T A^T \leq -b^T$. Renaming x y and rewriting yields the primal problem.

If the primal is in standard form, the constraints are equalities instead of inequalities, so multiplying by a negative number is perfectly acceptable. Therefore, the dual of a standard form has no restriction on whether or not π is non-negative. The slides show how to take the dual of a general linear program (not necessarily in canonical or standard form).

Now that we know how to systematically find the best possible lower bound using this technique, a natural question is how close to the optimal objective of the primal the lower bounds can get. It turns out that dual and the primal have the same objective value (strong duality)! To prove this, recall that $\bar{c}^T = c^T - c_B^T A_B^{-1} A$ and at optimality $\bar{c}^T \geq 0$. This implies $(c_B^T A_B^{-1})A \leq c^T$, so $\pi^T = c_B^T A_B^{-1}$ is dual feasible. The corresponding dual objective value is $\pi^T b = c_B^T A_B^{-1} b = c_B^T x_B^0 = c^T x^0$. Not only are the objective values of the primal and the dual equal, but a solution to the primal yields an immediate solution to the dual. Since the dual of the dual is the primal, a solution to the dual yields an immediate solution to the primal as well. This is the idea behind the dual simplex algorithm.

For x and π that are optimal solutions to the primal and dual respectively, $\pi^T b = c^T x$ so the inequalities $\pi^T b \leq \pi^T A x \leq c^T x$ in the proof of weak duality must holds with equality, so $\pi^T(Ax - b) = 0$ and $(c^T - \pi^T A)x = 0$. These are called the complementary slackness conditions. Note that the four vectors π, x, $Ax - b$ and $c^T - \pi^T A$ have non-negative components, so the dot products can only be zero if for every component i $\pi_i(Ax - b)_i = 0$ and $(c^T - \pi^T A)_i x_i = 0$.