ZOR – Methods and Models of Operations Research (1993) 37:171–186

Linear Programming with Entropic Perturbation[1]

S. C. FANG

North Carolina State University, Operations Research Program & Industrial Engineering Department, Raleigh, NC 27695-7913, USA

H. S. J. TSAO

University of California, Institute of Transportation Studies, Berkeley, CA 94720, USA

Abstract: In this paper, we derive an unconstrained convex programming approach to solving standard form linear programs through an entropic perturbation. The whole duality theory is established by using only one simple inequality "$\ln z \leq z - 1$ for $z > 0$". A curved search algorithm is also proposed for obtaining a pair of primal and dual ε-optimal solutions. The proposed algorithm is proven to be globally convergent with a quadratic rate of convergence. Computational results are included in support of theoretic findings.

1 Introduction

Since N. Karmarkar introduced his projective scaling algorithm [15] in 1984, various interior-point methods [12] have been proposed to compete with the classic simplex method [2]. Among many new research directions, Fang [8] recently proposed an unconstrained convex programming approach to solving linear programming problems in Karmarkar's form and suggested a careful review of the unconstrained convex optimization techniques [4, 11] for linear programming. The original work was done in a framework of geometric programming [17], and was later refined by a much simpler derivation [18]. The new approach is neither simplex based nor interior point based. It only involves solving an unconstrained convex programming dual problem and converting a dual solution to a primal ε-optimal solution.

In this paper, by using only one simple inequality "$\ln z \leq z - 1$, for $z > 0$," we further extend the unconstrained convex programming approach to solving standard form linear programming problems. Moreover, based on the curved search methods [1], we propose a quadratically convergent global algorithm for linear programming. Both theoretic proof and computational experience are included.

We shall introduce the unconstrained convex dual approach with entropic perturbation in Section 2, find an ε-optimal primal solution in Section 3, propose

[1] The work is partially supported by the North Carolina Supercomputing Center, the Cray Research Award, and the National Science Council of the Republic of China # NSC 81-0415-E-007-10.

a curved search algorithm in Section 4, prove its convergence properties in Section 5, discuss implementation issues in Section 6, report computational experience in Section 7, and conclude the paper by Section 8.

2 Unconstrained Convex Approach with Entropic Perturbation

Consider the following (primal) linear program in its standard form:

Program P: Minimize $\mathbf{c}^T\mathbf{x}$

$$\text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \tag{1.a}$$

$$\mathbf{x} \geq 0 \ , \tag{1.b}$$

where $\mathbf{c}$ and $\mathbf{x}$ are n-dimensional column vectors, $\mathbf{A}$ is an $m \times n$ $(m \leq n)$ matrix, $\mathbf{b}$ is an m-dimensional column vector, and 0 is the n-dimensional zero column vector.

The linear dual of Program P is given as follows:

Program D: Maximize $\mathbf{b}^T\mathbf{w}$

$$\text{subject to} \quad \mathbf{A}^T\mathbf{w} \leq \mathbf{c} \tag{2.a}$$

$$\mathbf{w} \in R^m \ . \tag{2.b}$$

Following the approach developed in [8], for any given scalar $\mu > 0$, instead of solving Program P directly, we consider a nonlinear program with an entropic perturbation, namely,

Program P_μ: Minimize $\mathbf{c}^T\mathbf{x} + \mu \sum_{j=1}^{n} x_j \ln x_j$

$$\text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \tag{3.a}$$

$$\mathbf{x} > 0 \ . \tag{3.b}$$

Notice that the root of this entropic approach can be traced back to references [5, 6, 7, 13, 14]. Also note that the entropic function $x \ln x$ is a strictly convex function well-defined on $[0, \infty)$ with a unique minimum value of $-1/e$ at $x = 1/e$, where $e = 2.718\ldots$.

Like all interior-point methods, we assume that Program P has an interior feasible solution $\mathbf{x} > 0$ (i.e., the Interior-Point Assumption). Under this assumption, Program P_μ is feasible for any $\mu > 0$. Moreover, since $0 \ln 0 = 0$, $c_j x_j + x_j \ln x_j \to +\infty$ as $x_j \to \infty$, and $x_j \ln x_j$ is strictly convex over its domain, for each j, we know that Program P_μ achieves a finite minimum at a unique point $\mathbf{x}^* \in R^n$ for each $\mu > 0$. More interestingly, as discussed in [8], if Program P also has a bounded feasible domain (i.e., the Bounded Feasible Domain Assumption), then as $\mu \to 0$ the optimal solution of Program P_μ approaches an optimal solution of Program P.

Superficially, Program P_μ seems to be more complicated than Program P because of the involvement of the nonlinear entropic function. In reality, this nonlinearity is the key to deriving an unconstrained dual convex program. To achieve this goal, consider the following simple inequality:

$$\ln z \le z - 1 , \qquad \text{for } z > 0 . \tag{4}$$

This inequality can be easily verified by its graph and notice that inequality (4) becomes an equality if and only if $z = 1$.

Now, for any $\mu > 0$, $w_i \in R$ $(i = 1, \ldots, m)$, and $x_j > 0$ $(j = 1, \ldots, n)$, we define

$$z_j = \frac{e^{[(\sum_{i=1}^m a_{ij} w_i - c_j)/\mu] - 1}}{x_j} , \qquad \text{for } j = 1, \ldots, n . \tag{5}$$

Since $x_j > 0$, $z_j > 0$ is implied. By (4), we have

$$\left[\left(\sum_{i=1}^m a_{ij} w_i - c_j\right)\Big/ \mu\right] - 1 - \ln x_j \le \frac{e^{[(\sum_{i=1}^m a_{ij} w_i - c_j)/\mu] - 1}}{x_j} - 1 .$$

Consequently,

$$x_j\left[\left(\sum_{i=1}^m a_{ij} w_i - c_j\right)\Big/ \mu\right] - e^{[(\sum_{i=1}^m a_{ij} w_i - c_j)/\mu] - 1} \le x_j \ln x_j .$$

By multiplying both sides by μ and summing over j, we have

$$\sum_{j=1}^n x_j\left(\sum_{i=1}^m a_{ij} w_i\right) - \mu \sum_{j=1}^n e^{[(\sum_{i=1}^m a_{ij} w_i - c_j)/\mu] - 1} \le \sum_{j=1}^n c_j x_j + \mu \sum_{j=1}^n x_j \ln x_j .$$

If x_j ($j = 1, \ldots, n$) satisfies $\sum_{j=1}^{n} a_{ij}x_j = b_i$ for $i = 1, 2, \ldots, m$, (i.e. $\mathbf{Ax} = \mathbf{b}$), then

$$\sum_{j=1}^{n} x_j \left(\sum_{i=1}^{m} a_{ij}w_i \right) = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} a_{ij}x_j \right) w_i = \sum_{i=1}^{m} b_i w_i \ .$$

Therefore, for any $\mathbf{x} > 0$ such that $\mathbf{Ax} = \mathbf{b}$, we know that

$$\sum_{i=1}^{m} b_i w_i - \mu \sum_{j=1}^{n} e^{[(\sum_{i=1}^{m} a_{ij}w_i - c_j)/\mu] - 1} \le \sum_{j=1}^{n} c_j x_j + \mu \sum_{j=1}^{n} x_j \ln x_j \ . \tag{6}$$

Recall that the right-hand side of (6) is the objective function of Program P_μ. Hence we define a geometric dual program D_μ as follows:

Program D_μ: Maximize $\quad d_\mu(\mathbf{w}) \equiv \sum_{i=1}^{m} b_i w_i - \mu \sum_{j=1}^{n} e^{[(\sum_{i=1}^{m} a_{ij}w_i - c_j)/\mu] - 1}$

subject to $\mathbf{w} \in R^m$.

Note that Program D_μ is an unconstrained problem and the sum in each of the n exponents in the second term of its objective function is simply proportional to the amount of violation of the corresponding constraint of Program D. However, this nonlinear term of $d_\mu(\mathbf{w})$ is neither a penalty function nor a barrier function in the traditional sense, since the traditional penalty function does not impose penalty on any feasible solution and the barrier function ensures that the objective value approaches infinity at the boundary of feasible region. More importantly, if Program D_μ attains a finite optimum at $\mathbf{w}^*(\mu)$, then $\mathbf{w}^*(\mu)$ becomes a feasible solution to Program D as μ approaches 0. Actually, it solves Program D as μ approaches 0.

Several observations can be made here.

Observation 1: Since inequality (6) holds for any $\mathbf{w} \in R^m$ and $\mathbf{x} \in R^n$ such that $\mathbf{Ax} = \mathbf{b}$ and $\mathbf{x} > 0$, therefore we have the following "weak duality theorem":

Theorem 1: $\mathrm{Min}(P_\mu) \ge \mathrm{Max}(D_\mu)$.

Observation 2: Recall that inequality (4) becomes an equality if and only if $z = 1$. Hence inequality (6) becomes an equality if and only if

$$z_j = \frac{e^{[(\sum_{i=1}^{m} a_{ij}w_i - c_j)/\mu] - 1}}{x_j} = 1$$

or, equivalently,

$$x_j = e^{[(\sum_{i=1}^m a_{ij}w_i - c_j)/\mu] - 1} .$$

Consequently, we have the following theorem:

Theorem 2: Given that $\mathbf{w}^* \in R^m$ and $\mathbf{x}^* \in R^n$ such that $\mathbf{A}\mathbf{x}^* = \mathbf{b}$ and $\mathbf{x}^* > 0$, if

$$x_j^* = e^{[(\sum_{i=1}^m a_{ij}w_i^* - c_j)/\mu] - 1} , \qquad \text{for } j = 1, \ldots, n , \tag{7}$$

then $\mathbf{x}^*$ is an optimal solution to Program P_μ and $\mathbf{w}^*$ is an optimal solution to Program D_μ. Moreover, $\text{Min}(P_\mu) = \max(D_\mu)$.

Observation 3: The (k_1, k_2)-th element of the Hessian matrix of function $d_\mu(\mathbf{w})$ is given by

$$\frac{\partial d_\mu(\mathbf{w})}{\partial w_{k_1} \partial w_{k_2}} = -\frac{1}{\mu} \sum_{j=1}^n e^{[(\sum_{i=1}^m a_{ij}w_i - c_j)/\mu] - 1} a_{k_1 j} a_{k_2 j} .$$

Therefore, the Hessian matrix can be written as $\mathbf{A}\mathbf{D}_\mathbf{r}(\mathbf{w})\mathbf{A}^T$, where $\mathbf{D}_\mathbf{r}(\mathbf{w})$ is an $n \times n$ diagonal matrix with $\mathbf{r}_j(\mathbf{w})$ as its j-th diagonal element and

$$\mathbf{r}_j(\mathbf{w}) = -\frac{1}{\mu} e^{[(\sum_{i=1}^m a_{ij}w_i - c_j)/\mu] - 1} < 0 .$$

By matrix theory, the Hessian matrix must be nonsingular and negative definite as long as $\mathbf{A}$ has full row-rank. In this case, we know that $d_\mu(\mathbf{w})$ is strictly concave and we have the following result:

Theorem 3: If the constraint matrix $\mathbf{A}$ in Program P has full row-rank, then Program D_μ has a strictly concave objective function $d_\mu(\mathbf{w})$.

Observation 4: Under the Interior-Point Assumption, Program P (hence P_μ) has an interior feasible solution. From convex analysis (Fenchel's Theorem) [17, 19], we know that there is no duality gap between the pair of Programs P_μ and D_μ. Recall that Program P_μ always achieves a finite optimum as long as $\mu > 0$. Therefore, if $\mathbf{A}$ has full row-rank, then $D_\mu(\mathbf{w})$ is strictly concave and Program D_μ must also achieve a finite optimum at a unique maximizer $\mathbf{w}^*(\mu) \in R^m$. Now, since $d_\mu(\mathbf{w})$ is continuously differentiable, the first order optimality conditions hold at $\mathbf{w}^*(\mu)$. In other words, setting $\nabla d_\mu(\mathbf{w}^*(\mu)) = 0$, we have

$$\sum_{j=1}^{n} e^{[(\sum_{i=1}^{m} a_{ij} w_i^*(\mu) - c_j)/\mu] - 1} a_{kj} = b_k \ , \qquad \text{for } k = 1, 2, \ldots, m \ .$$

If we further define $\mathbf{x}^*(\mu) \in R^n$ according to (7), then the above equation becomes $\mathbf{A}\mathbf{x}^*(\mu) = \mathbf{b}$. Hence Theorem 2 further implies the following "strong duality theorem":

Theorem 4: If Program P has an interior feasible solution and a full row-rank constraint matrix $\mathbf{A}$, then Program D_μ has a unique optimal solution $\mathbf{w}^*(\mu) \in R^m$. In this case, formula (7) provides a dual-to-primal conversion which defines the optimal solution $\mathbf{x}^*(\mu)$ of Program P_μ. Moreover, $\text{Min}(\mathbf{P}_\mu) = \text{Max}(D_\mu)$.

Note that the above strong duality theorem can also be viewed as a special case of the Fenchel duality theorem in separable convex programming, but it is derived in an innovative way which tackles our problem directly. Throughout this paper, we hold the Interior-Point Assumption and assume that matrix $\mathbf{A}$ has full row-rank. Besides, $\mathbf{x}^*(\mu)$ and $\mathbf{w}^*(\mu)$ will be used to denote the unique optimal solution of P_μ and D_μ respectively.

3 An ε-Optimal Solution

Since Program P_μ and Program P have identical feasible domain, any solution to Program P_μ must be feasible to Program P. For any given $\varepsilon > 0$, our objective is to find a sufficiently small $\mu > 0$ such that $\mathbf{x}^*(\mu)$ is an ε-optimal solution to Program P.

To achieve our goal, we first study the behavior of $\mathbf{x}^*(\mu)$ as μ becomes smaller. Suppose that $\mathbf{x}^1$ solves Program P_{μ_1} and $\mathbf{x}^2$ solves Program P_{μ_2}, where $\mu_1 > \mu_2 > 0$. Then, since $\mathbf{x}^1$ is an optimal solution to Program P_{μ_1}, we have

$$\sum_{j=1}^{n} c_j x_j^1 + \mu_1 \sum_{j=1}^{n} x_j^1 \ln x_j^1 \leq \sum_{j=1}^{n} c_j x_j^2 + \mu_1 \sum_{j=1}^{n} x_j^2 \ln x_j^2 \ . \tag{8}$$

Similarly, for $\mathbf{x}^2$ solves Program P_{μ_2}, we see

$$\sum_{j=1}^{n} c_j x_j^2 + \mu_2 \sum_{j=1}^{n} x_j^2 \ln x_j^2 \leq \sum_{j=1}^{n} c_j x_j^1 + \mu_2 \sum_{j=1}^{n} x_j^1 \ln x_j^1 \ . \tag{9}$$

Multiplying (9) by -1 and adding the resulting inequality to (8), we obtain

$$(\mu_1 - \mu_2) \sum_{j=1}^{n} x_j^1 \ln x_j^1 \le (\mu_1 - \mu_2) \sum_{j=1}^{n} x_j^2 \ln x_j^2 \ .$$

Since $\mu_1 > \mu_2$, we know that

$$\sum_{j=1}^{n} x_j^1 \ln x_j^1 \le \sum_{j=1}^{n} x_j^2 \ln x_j^2 \ . \tag{10}$$

After rearranging terms in (9) and using (10), we further have

$$0 \le \mu_2 \left(\sum_{j=1}^{n} x_j^2 \ln x_j^2 - \sum_{j=1}^{n} x_j^1 \ln x_j^1 \right) \le \sum_{j=1}^{n} c_j x_j^1 - \sum_{j=1}^{n} c_j x_j^2 \ .$$

Rearranging terms in (8), we obtain

$$\sum_{j=1}^{n} c_j x_j^1 - \sum_{j=1}^{n} c_j x_j^2 \le \mu_1 \left(\sum_{j=1}^{n} x_j^2 \ln x_j^2 - \sum_{j=1}^{n} x_j^1 \ln x_j^1 \right) .$$

Combining the above two inequalities together gives the following theorem:

Theorem 5: As $\mu > 0$ goes to 0, $\mathbf{c}^T \mathbf{x}^*(\mu)$ decreases monotonically. Moreover, for $\mu_1 > \mu_2 > 0$, if $\mathbf{x}^1$ and $\mathbf{x}^2$ solves Programs P_{μ_1} and P_{μ_2} respectively, then

$$\begin{aligned} 0 &\le \mu_2 \left(\sum_{j=1}^{n} x_j^2 \ln x_j^2 - \sum_{j=1}^{n} x_j^1 \ln x_j^1 \right) \le \mathbf{c}^T \mathbf{x}^1 - \mathbf{c}^T \mathbf{x}^2 \\ &\le \mu_1 \left(\sum_{j=1}^{n} x_j^2 \ln x_j^2 - \sum_{j=1}^{n} x_j^1 \ln x_j^1 \right) . \end{aligned} \tag{11}$$

Now, in addition to the Interior-Point Assumption, we add the Bounded Feasible Domain Assumption. Under this circumstance, it is not difficult to see that Program P has an optimal solution $\mathbf{x}^*$ and $\mathbf{x}^*(\mu) \to \mathbf{x}^*$ as $\mu \to 0$. Moreover, there exists a large positive number $M > 0$ such that the primal feasible domain (1.a&b) is contained in the spheroid centered at the origin with a radius of M. Thus, for any primal feasible solution $\mathbf{x}$, we have

$$|x_j \ln x_j| \le \tau \equiv \max\{1/e, |M \ln M|\} \ , \qquad \text{for } j = 1, \ldots, n \ . \tag{12}$$

Consequently, by taking $\mu_2 \to 0$ in (11), we have

$$|\mathbf{c}^T\mathbf{x}^1 - \mathbf{c}^T\mathbf{x}^*| \leq \mu_1(n\tau + n\tau) = 2n\tau\mu_1 \ .$$

Therefore, for any given $\varepsilon > 0$, if we choose

$$\mu_1 = \varepsilon/2n\tau \ , \tag{13}$$

then $|\mathbf{c}^T\mathbf{x}^1 - \mathbf{c}^T\mathbf{x}^*| \leq \varepsilon$. In other words, $\mathbf{x}^1$ is an ε-optimal solution.

In summary, we have the following result:

Theorem 6: Under the Interior-Point and Bounded Feasible Domain Assumptions, if $\mu > 0$ is chosen according to (13), then the optimal solution to Program P_μ is an ε-optimal solution to Program P.

A couple of observations can be made here.

Observation 5: For a linear program with bounded feasible domain, according to den Hertog et al. [3], by appropriate scaling, it is equivalent to a linear programming problem with all variables bounded above by 1. In this case, since $-1/e \leq x \ln x \leq 0$ for $0 \leq x \leq 1$, we see that $\tau = 1/e$ and $\mu = \varepsilon e/2n$ is good enough for any linear program scaled according to den Hertog et al.

Observation 6: Note that the magnitude of the right-most term of (11) is proportional to the difference of entropy evaluated at two different points, one of which is to be used as an approximation of the other. The bound provided in Theorem 6 is calculated based on the triangular inequality and hence may be too conservative. In other words, for a practical implementation, μ may be chosen much bigger than the value specified by (13).

4 A Curved Search Algorithm

We now outline the unconstrained convex programming approach to solving standard form linear programs under the assumptions of interior point and bounded feasible domain:

Step 1: Given $\varepsilon > 0$, compute μ according to formula (13).

Step 2: Solve Program D_μ by unconstrained convex optimization techniques for an optimal solution $\mathbf{w}^*(\mu)$.

Step 3: Compute the optimal solution $\mathbf{x}^*(\mu)$ of Program P_μ according to formula (7). Then $\mathbf{x}^*(\mu)$ is an ε-optimal solution to Program P.

It is clear that Step 2 is the major source of required computation. It was suggested in [8] to closely investigate and customize unconstrained optimization methods for this purpose. In this section we customize the "curved search method" proposed by Ben-Tal et al. [1] to achieve global convergence with a quadratic rate of convergence.

Instead of solving Program D_μ directly, we solve the following equivalent convex minimization problem:

Program D'_μ: Minimize $f(\mathbf{w}) \equiv -d_\mu(\mathbf{w})$

Subject to $\mathbf{w} \in \mathrm{R}^m$

As discussed before, when matrix $\mathbf{A}$ is of full row-rank, $f(\mathbf{w})$ is a strictly convex and twice continuously differentiable function over R^m with

$$\frac{\partial f}{\partial w_{k_1}} = \sum_{j=1}^{n} e^{[(\sum_{i=1}^m a_{ij}w_i - c_j)/\mu] - 1} a_{k_1 j} - b_{k_1} \ , \tag{14}$$

and

$$\frac{\partial f}{\partial w_{k_1} \partial w_{k_2}} = \frac{1}{\mu} \sum_{j=1}^{n} e^{[(\sum_{i=1}^m a_{ij}w_i - c_j)/\mu] - 1} a_{k_1 j} a_{k_2 j} \ . \tag{15}$$

The Hessian matrix $\mathbf{H}$ is positive definite and can be written as $\mathbf{A}\mathbf{D}_s(\mathbf{w})\mathbf{A}^T$, where $\mathbf{D}_s(\mathbf{w})$ is an $n \times n$ diagonal matrix with $s_j(\mathbf{w})$ as its j-th diagonal element such that

$$s_j(\mathbf{w}) = -\mathbf{r}_j(\mathbf{w}) = \frac{1}{\mu} e^{[(\sum_{i=1}^m a_{ij}w_i - c_j)/\mu] - 1} > 0 \ . \tag{16}$$

The basic idea of the curved search method is to improve a current solution by moving along a quadratic curve, while most classical iterative methods move along a straight line. More precisely, for an unconstrained convex minimization problem with a twice continuously differentiable objective function f, the curved-search method moves from one solution $\mathbf{w}_k$ to the next solution $\mathbf{w}_{k+1}$ along the quadratic curve (in variable t)

$$q_k(t) = w_k + td_k + \frac{1}{2}t^2 z_k \qquad (\text{for } t \geq 0) \ ,$$

where

$$d_k = -\beta_k \frac{|\nabla f(\mathbf{w}_k)|^2}{(\nabla f(\mathbf{w}_k))^T[\nabla^2 f(\mathbf{w}_k)]^{-1}\nabla f(\mathbf{w}_k)}[\nabla^2 f(\mathbf{w}_k)]^{-1}\nabla f(\mathbf{w}_k) \qquad \text{and}$$

$$z_k = -\alpha_k |\nabla f(\mathbf{w}_k)| \nabla f(\mathbf{w}_k) \ .$$

Searching for an appropriate step-length t_k such that

$$t_k \in \arg\min_{t>0} f\left(\mathbf{w}_k + t\mathbf{d}_k + \frac{1}{2}t^2\mathbf{z}_k\right) \tag{17}$$

results in a new solution $\mathbf{w}_{k+1}$ defined by

$$\mathbf{w}_{k+1} = w_k + t_k d_k + \frac{1}{2}t_k^2 z_k \ . \tag{18}$$

Notice that α_k and β_k are adjustable positive parameters which could be fine-tuned for better performance of particular problems. Also notice that, as pointed out in [1], when $\alpha_k \equiv 0$, the curved-search algorithm becomes the (signed) Newton method; and, it becomes the steepest descent algorithm when $\beta_k \equiv 0$.

Based on the above curved search method, we propose the following algorithm for solving Program D'_μ and refer to it as the CS-LPE (Curved Search for Linear Programs with Entropic perturbation) algorithm.

The CS-LPE Algorithm

Initialization: Choose sufficiently small real numbers $\zeta > 0$, $\delta > 0$ and $\gamma > 0$.
Choose a positive sequence $\{\alpha_k > 0\}$ with a finite limit superior.
Choose a positive sequence $\{\beta_k > 0\}$ with a finite limit superior and a positive limit inferior.
Set $k = 0$ and select any starting solution $\mathbf{w}_0 \in R^m$.

Iteration: Compute $\mathbf{g}_k \equiv \nabla f(\mathbf{w}_k)$ with its k_1-th element defined by Eq. (14).
Compute $\mathbf{H}_k \equiv \nabla^2 f(\mathbf{w}_k)$ with its (k_1, k_2)-th element defined by Eq. (15).
Stop, if $|\mathbf{g}_k| \le \zeta$. Otherwise, proceed.
Solve $\mathbf{H}_k\mathbf{v}_k = \mathbf{g}_k$ for $\mathbf{v}_k$.

```
Compute $G_k \equiv \mathbf{g}_k^T \mathbf{v}_k$, $\gamma_k \equiv \frac{|G_k|}{|\mathbf{v}_k||\mathbf{g}_k|^2}$ and $\delta_k \equiv (\det \mathbf{H}_k)^2$ .
If $\delta_k > \delta$ and $\gamma_k > \gamma$, then
    Compute $\mathbf{d}_k \equiv -\beta_k \frac{|\mathbf{g}_k|^2}{G_k} \mathbf{v}_k$ and $\mathbf{z}_k \equiv -\alpha_k |\mathbf{g}_k| \mathbf{g}_k$.
Otherwise,
    Set $\mathbf{d}_k = -\mathbf{g}_k$ and $\mathbf{z}_k = 0$.
Compute a step-length $t_k \in \arg\min_{t>0} f\left(\mathbf{w}_k + t\mathbf{d}_k + \frac{1}{2}t^2\mathbf{z}_k\right)$ .
Iterate $\mathbf{w}_{k+1} = \mathbf{w}_k + t_k\mathbf{d}_k + \frac{1}{2}t_k^2\mathbf{z}_k$.
Reset $k \to k+1$ for the next iteration.
```

5 Convergence Properties

In this section we show that, under some mild conditions, the CS-LPE algorithm indeed produces a sequence of iterates that converge to the unique optimal solution of Program D'_μ with a quadratic rate of convergence. Actually this result is expected, since the curved search method is a combination of the signed Newton method (which has local quadratic rate of convergence) and the steepest descent method (which converges globally).

The following two theorems state these facts.

Theorem 7: If Program P has an interior feasible solution and a full row-rank constraint matrix $\mathbf{A}$, then the CS-LPE algorithm either stops after finitely many steps or generates an infinite sequence $\{\mathbf{w}_k \in R^m | k = 1, 2, \ldots\}$ such that $f(\mathbf{w}_{k+1}) < f(\mathbf{w}_k)$ for each k; $\{\mathbf{w}_k \in R^m | k \varepsilon N\}\}$ has at least a cluster point $\mathbf{w}^c$ in the level set $L_0 \equiv \{\mathbf{w} \in R^m | f(\mathbf{w}) \leq f(\mathbf{w}_0)\}$; for each such cluster point $\mathbf{w}^c$, $|\nabla f(\mathbf{w}^c)| \leq \zeta$.

Proof: Recall that the objective function f of Program D'_μ is twice continuously differentiable. According to Theorem 3.1 of [1], if we can further show that the level set L_0 is compact, then the proof is complete.

Under current assumptions, by Theorem 4, we know that f has a unique minimizer $w^* \in R^m$ with a finite minimum value. Therefore, each level set of f, including L_0, is compact. □

Note that, under the assumptions of Theorem 7, since f has a unique optimal solution $\mathbf{w}^*$ satisfying the first order optimality conditions, the infinite sequence generated by the CS-LPE algorithm must converge to the unique cluster point $\mathbf{w}^c = \mathbf{w}^*$.

Theorem 8: Under the assumptions of Theorem 7, the infinite sequence $\{\mathbf{w}_k \in R^m | k\varepsilon N\}$ generated by the CS-LPE algorithm converges to $\mathbf{w}^*$ at a quadratic rate.

Proof: Based on Theorem 4.4 of [1], to establish the quadratic rate of convergence for the CS-LPE algorithm, we need to show that the following four conditions are satisfied: (i) $\{\mathbf{w}_k \in R^m | k\varepsilon N\}$ converges to $\mathbf{w}^*$, (ii) $\nabla f(\mathbf{w}^*) = 0$, (iii) $\nabla^2 f(\mathbf{w}^*)$ is positive definite, and (iv) $\mathbf{H}(\cdot) \equiv \nabla^2 f(\cdot)$ is locally Lipschitz-continuous in a neighborhood of $\mathbf{w}^*$.

The first three conditions are clearly met from our previous discussions. As to condition (iv), recall that $\mathbf{H} = \mathbf{A}\mathbf{D}_s(\mathbf{w})\mathbf{A}^T$ and $\mathbf{D}_s(\mathbf{w})$ is a diagonal matrix with $s_j(\mathbf{w})$ as its j-th diagonal element. From Equation (16), we know that $s_j(\mathbf{w}), j = 1, 2, \ldots, n$, belongs to C^∞. Therefore, every element of $\mathbf{H}$ is a finite combination of C^∞ functions. Since differentiability implies Lipschitz continuity, we know that condition (iv) is satisfied too and the proof is complete. □

6 Implementation Issues

Before we report our computational experience with the CS-LPE algorithm, we discuss four related implementation issues:

1. In each iteration of the CS-LPE algorithm, major computations include solving $\mathbf{H}_k\mathbf{v}_k = g_k$ for $\mathbf{v}_k$ and finding t_k by minimizing $f(\mathbf{w}_k + t\mathbf{d}_k + \frac{1}{2}t^2\mathbf{z}_k)$ over $t > 0$.

To calculate $\mathbf{v}_k$ in our implementation, we chose the Gaussian Elimination method because our main interest is in investigating the total number of iterations. Clearly, the more efficient Cholesky factorization method may be applied to solving this system of equations involving a symmetric positive-definite matrix. For the same reason, we chose a commonly-used line-search procedure, consisting of the search for a three-point pattern and the Golden-Section search [16], to determine the step-length t_k.

2. Consider Program D'_μ again. Evaluation of exponential functions is required in calculating f, ∇f, and $\nabla^2 f$. By examing the definition of f and equations (14) and (15), it is clear that we have to overcome the possible arithmetic overflow problem, in particular when μ is very small.

To prevent the overflow problem from happening, notice that Program D'_μ is a minimization problem and the exponential terms contained in the gradient vector and the Hessian matrix are identical to those contained in the objective function. Therefore, if an initial solution $\mathbf{w}_0$ is properly chosen such that the initial objective value is good for a floating point representation, then there is no overflow problem in evaluating the exponential functions subsequently. Technically, this can be achieved by choosing $\mathbf{w}_0$ to be feasible to Program D, since for such a solution all the exponents involved in the objective function are strictly negative no matter how small μ is. Moreover, in searching for the three-point pattern, we need to start at a point that is close to the current solution. With these two precautionary steps, the overflow problem never occurred in our implementation.

3. Since an exponential function belongs to the category of transcendental functions, a significant amount of computational effort is needed in evaluating f, ∇f, and $\nabla^2 f$. Realizing the fact that there are n exponential functions involved in the gradient vector and $m^2 n$ exponential functions involved in the Hessian matrix, one might think that the CS-LPE algorithm would require excessive amount of computation. Fortunately, this is not the case. In our implementation, only n "seed exponential functions" need to be evaluated in each iteration. They are the diagonal elements $s_j(\mathbf{w})$ ($j = 1, \ldots, n$) of the diagonal matrix $\mathbf{D}_s(\mathbf{w})$ defined by Equation (16). Once these n seeds are computed and stored, both the gradient vector and Hessian matrix can be computed via simple arithmetic operations. Therefore, no excessive computational effort is required. Also, because the total number of these seeds grows linearly with n, the number of primal variables, the computational time spent on these calculations accounts for a less significant portion in solving large-size problems.

4. Once the n seed exponential functions are calculated and stored, it takes $O(m^2 n)$ elementary operations to compute the Hessian matrix and $O(n^3)$ (or fewer) elementary operations for matrix inversion. This is the real computational bottleneck for solving large-size problems.

Since every element of the Hessian matrix $\mathbf{H}_k$ can be independently calculated, an implementation with parallel computation can certainly improve the performance of the CS-LPE algorithm. Moreover, the use of sparse matrix techniques can also lead to more efficient implementation for solving large-size problems. But it is beyond the scope of this paper.

7 Computational Experience

In this section, we report our computational experience with the CS-LPE algorithm. The testing problems were randomly generated with their sizes ranging from 30 variables with 10 constraints to 4,000 variables with 1,000 constraints. All testing problems have a full-row rank constraint matrix $\mathbf{A}$ and satisfy both the Interior-Point and Bounded Feasible Domain assumptions.

We briefly describe how an $m \times n$ standard-form linear program is "randomly" generated for the test. Our test linear program contains $m - 1$ "regular" constraints and one bounding constraint. The regular constraints are randomly generated as inequality constraints first in the $n - m$-dimensional vector space and then converted to equality ones in the n-dimensional space. The coefficients on the left-hand-side of the inequalities are randomly generated between -1 and 1. The right-hand-side of each of these inequalities is chosen so that (i) the associated $n - m$-dimensional hyperplane intersects the line segment connecting $(0, 0, \ldots, 0)$ and $(1, 1, \ldots, 1)$ and (ii) the point of intersection is uniformly distributed over this line segment. The sign of the inequality is chosen so that the point $(0.5, 0.5, \ldots, 0.5)$ is feasible. The bounding constraint is set to be $\sum_{j=1}^{n} x_j \leq M$ for a large constant M. All elements of the cost vector $\mathbf{c}$ are randomly generated between 0 and 1. In this way, the m-dimensional zero vector 0 is feasible to Program D and hence an initial solution for the CS-LPE algorithm which avoids the otherwise potential problem of arithmetic overflow.

To demonstrate the computational procedure, regardless of problem size, we choose $\mu = 0.001$ in the tests. For simplicity reason, we always set $\alpha_k = \beta_k = 1$ and $\zeta = 10^{-5}$. The line search stops when the difference of the objective values of two consecutive test points is within 10^{-10}, and the CS-LPE algorithm stops when $|g_k| \leq \zeta = 10^{-5}$. To avoid arithmetic overflow problem in our experiment, the initial solution $\mathbf{w}_0$ is chosen to be dual feasible.

We ran the CS-LPE algorithm on a SUN SPARCII workstation for all test problems except the 1000×4000 problem, for which we used an Alliant FX8 with its parallel/vector processing capability. Table 1 summarizes our computational experience. The six columns correspond to problem size, the number of test problems for that size, the average number of iterations for the test problems of that size, CPU time spent on Hessian calculations, CPU time spent on Hessian inversion, and CPU for the rest of the calculations.

Table 1: CS-LPE Algorithm for Randomly-Generated Problems

Size	#Prob.	#Iter.	CPU%-Hessian	CPU%-Inversion	CPU%-Other
10×30	4	14.3	18%	11%	71%
25×100	4	35.5	36%	19%	45%
50×150	4	45.0	43%	32%	25%
100×300	4	57.0	47%	37%	16%
300×1000	4	64.3	54%	37%	9%
500×1500	4	80.3	55%	41%	4%
1000×4000	1	94.0	NA*	NA*	NA*

NA*: Not included for comparison because parallel/vector processing was invoked.

Some important facts can be added to Table 1:

(i) According to our experience, the CS-LPE algorithm always converges as it should.
(ii) In order to verify the quadratic rate of convergence, we calculated $r_k \equiv |\mathbf{w}_k - \mathbf{w}^*|/|\mathbf{w}_{k-1} - \mathbf{w}^*|^2$. In our experiment, r_k indeed converges up until the current solution is in an "immediate" neighborhood of the optimal solution $\mathbf{w}^*$. Then the ratio becomes sensitive to numerical accuracy. This observation together with the low number of iterations could confirm the quadratic rate of convergence of the CS-LPE algorithm.
(iii) By setting the initial solution at a point feasible to Program D and choosing appropriate first step-length in the three-point search, no arithmetic overflow problem ever occurred in our experiment.
(iv) Observe that the percentage of CPU-Other decreases as problem size increases. This indicates that the computational effort spent in evaluating the exponential functions is relatively small in solving large-size problems.
(v) Table 1 also confirmed that the calculation of the Hessian matrix $\mathbf{H}$ and its inversion really dominates the computational complexity in solving large-size problems.
(vi) The number of total iterations grows slowly with the size of the problem.
(vii) For test problems with the same size, the number of iterations required by individuality varies only slightly.

8 Concluding Remarks

We have introduced an unconstrained convex programming approach to obtain an ε-optimal solution of standard form linear programs under the assumptions of interior point and bounded feasible domain. The ε-optimal solution is literally obtained by solving an unconstrained dual convex program. Conversion from the dual optimal solution to the ε-optimal solution is effortless. We also customized a curved search method for solving the unconstrained dual convex program. This algorithm converges globally with a quadratic rate of convergence.

Notice that the moving direction of the CS-LPE algorithm depends on its Hessian matrix $\mathbf{H} = \mathbf{A}\mathbf{D}_s(\mathbf{w})\mathbf{A}^T$ while the moving direction of any known interior-point algorithms [12] is determined by a positive definite matrix with exactly the same structure. This should lead to comparable computing time for one iteration among these different algorithms when applied to a common problem. According to our computational experience, the total number of iterations grows slowly with respect to the increase in problem size. This indicates the potential of this unconstrained convex programming approach for efficiently solving large-size linear programming problems. Finally, among other potential advantages, this approach does not require Phase-I iterations.

References

[1] Ben-Tal A, Melman A and Zowe J (1990) Curved Search Methods for Unconstrained Optimization. Optimization 21 (5):669–695

[2] Dantzig GB (1963) Linear Programming and Extensions. Princeton University Press, Princeton, NJ

[3] Den Hertog D, Roos C and Terlaky T (1991) Inverse Barrier Methods for Linear Programming. Delft University of Technology, Report of the Faculty of Technical Mathematics and Informatics, No. 91-27

[4] Dennis JE and Schnabel RB (1983) Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Prentice Hall, Englewood, NJ

[5] Erlander S (1981) Entropy in Linear Programs. Mathematical Programming 21:137–151

[6] Erickson JR (1981) Algorithms for Entropy and Mathematical Programming. PhD Thesis, Linkoping University, Sweden

[7] Erickson JR (1985) An Iterative Primal-Dual Algorithm for Linear Programming. Report LitH-MAT-R-1985-10, Department of Mathematics, Linkoping University, Sweden

[8] Fang SC (1992) A New Unconstrained Convex Programming Approach to Linear Programming. North Carolina State University Operations Research Report No. 243 (Feb. 1990), Zeischrift fur Operations Research 36:149–161

[9] Fang SC and Rajasekera JR (1989) Quadratically Constrained Minimum Cross-Entropy Analysis. Mathematical Programming 44:85–96

[10] Fang SC and Tsao JHS (1991) A Quadratically Convergent Global Algorithm for the Linearly-Constrained Minimum Cross-Entropy Problem. North Carolina State University Operations Research Report 255 to appear in EJOR

[11] Fiacco AV, McCormick GP (1968) Nonlinear Programming: Sequential Unconstrained Minimization Techniques. Wiley, New York

[12] Goldfarb D, Todd MJ (1988) Linear Programming. Cornell University, School of OR and IE, Tech. Report 777

[13] Guiasu S (1977) Information Theory with Applications. McGraw-Hill, New York

[14] Gill PE, Murray W, Saunders M, Tomlin TA and Wright MH (1986) On Projected Newton Barriers for Linear Programming and An Equivalence to Karmarkar's Projective Method. Mathematical Programming 36:183–209

[15] Karmarkar N (1984) A New Polynomial Time Algorithm for Linear Programming. Combinatorica 4:373–395

[16] Minoux M (1986) Mathematical Programming Theory and Algorithms. John Wiley & Sons

[17] Peterson EL (1976) Geometric Programming. SIAM Review 19:1–45

[18] Rajasekera JR and Fang SC (1991) On the Convex Programming Approach to Linear Programming. North Carolina State University Operations Research Report 245 (April 1990), Operations Research Letters 10:309–312

[19] Rockaffelar RT (1970) Convex Analysis. Princeton University Press, Princeton, New Jersey

Received: September 1992