Fast Algorithms for Convex Quadratic Programming and Multicommodity Flows

Sanjiv Kapoor and Pravin M. Vaidya[1]

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

In the first part of the paper, we extend Karmarkar's interior point method to give an algorithm for Convex Quadratic Programming which requires $O(N^{3.67}(\log L)(\log N)L)$ arithmetic operations. At each iteration, Karmarkar's method locally minimizes the linear (convex) numerator of a transformed objective function in the transformed domain. However, in the case of Convex Quadratic Programming the numerator of the transformed objective function is not necessarily convex. We give a method that, at each iteration, locally optimizes the original objective function in the original domain itself. As a consequence we also obtain a monotonic decrease in the objective function. In the second part, we show how to solve the linear program describing the multicommodity flow problem, with s commodities, in $O(s^{3.5}v^{2.5}e\,L)$ arithmetic operations. In each problem arithmetic operations are performed to a precision of $O(L)$ bits where L is bounded by the number of bits in the input.

1. Convex Quadratic Programming

1.1. Introduction

In the first part of the paper we consider the problem of minimizing convex quadratic functions over polytopes, i.e. the Quadratic programming problem,

$$\min \quad f(x) = \frac{1}{2}x^TBx + p^Tx$$
$$\text{s.t. } Ax \le b$$
$$x \ge 0$$

where p and x are in R^n, A is in $R^{m\times n}$, $b \in R^m$ and $B \in R^{n\times n}$ is a positive semi-definite matrix. Let $N = n+m$. This problem was first solved by adapting the simplex method for linear programming [10]. A polynomial time algorithm for this problem was first presented in [6]. This polynomial time algorithm uses the ellipsoid method and in the worst case performs $O((N^4L)$ arithmetic operations where each operation requires a precision of $O(L)$ bits. (L is bounded by the number of bits in the input). Here we describe an algorithm for the Quadratic programming problem which requires $O(N^{3.67}(\log L)(\log N)L)$ arithmetic operations each performed up to a precision of $O(L)$ bits where

$$L = \log\text{ (largest absolute value of the determinant of any square submatrix of } \begin{bmatrix} B & A^T \\ A & 0 \end{bmatrix})$$
$$+\log(\max_i p_i) + \log(\max_i b_i) + \log N$$

[1]The names of the authors are in alphabetical order. The 1st author was supported by NSF under grant NSF DCR-8404239 and the 2nd author was supported by a fellowship from the Shell Foundation.

We assume without loss of generality that the problem is in the standard form,

$$\min f(x) = \frac{1}{2}x^TBx + p^Tx$$
$$\text{s.t. } Ax + y = b$$
$$e^Tz = M$$
$$z \ge 0$$

where $z^T = [z_1, z_2, x^T, y^T]$, $N = n+m+2$, $z \in R^N$, $x \in R^n$, $y \in R^m$, $A \in R^{m\times n}$, $b \in R^m$ and $e^T=[1,1,...,1] \in R^N$. Any Quadratic programming problem can be transformed into the above form by a shift of origin, addition of slack variables y_i and by introducing the equality

$$\sum_{i=1}^{n} x_i + \sum_{i=1}^{m} y_i + z_1 + z_2 = M$$

for a large enough value of M. ($M = N^4 2^L$ suffices).

We follow Karmarkar's method for linear programming and at each iteration reduce the global optimization to a series of local optimizations. The point obtained after $O(NL)$ such iterations is sufficiently close to the optimal and an exact optimum is then found. As in [4] convergence is measured by a potential function. In order to obtain a reduction in the potential function, at each iteration Karmarkar uses a projective transformation to obtain a local optimization problem in the transformed domain. The projective transformation maps the linear objective function to a ratio of linear functions and the local optimization involves the minimization of the numerator of the transformed objective function over an ellipsoid. For the Quadratic programming problem however, since the objective function contains both a linear and a quadratic term the application of the projective transformation does not always map the convex quadratic form to a function whose numerator is convex. So the approach of locally optimizing the numerator of the transformed objective function fails. In section 1.2 we show that if the function $f(z)$ being minimized is convex, a potential decrease can still be found by a set of local optimizations in the original domain itself. In each local optimization $f(z)$

is minimized over an ellipsoid. As a consequence we obtain an algorithm where the value of the objective function also decreases at each iteration. In section 1.3 we then present an algorithm which minimizes convex functions over polytopes.

In section 1.4 we reduce the local optimization problems arising in the convex Quadratic programming problem to a simpler form which we show how to solve efficiently. In section 1.5 we improve the amortized complexity of the convex Quadratic programming algorithm to $O(N^{3.67}(\log N)(\log L)L)$ arithmetic operations by solving slightly modified local optimization problems. In section 1.6 we show how to find an exact optimum from a point sufficiently close in objective function value to the optimum. Finally, in section 1.7 it is shown that $O(L)$ bit numbers suffice for the arithmetic operations.

1.2. Convergence via local optimizations

We measure convergence by means of the potential function

$$P(z)=\sum_{i=1}^{N}\ln\left(\frac{f(z)-f_0}{z_i}\right)$$

where $f(z)$ is the function being optimized and f_0 is the minimum value of $f(z)$ over the polytope. We assume that $f(z)$ is convex. In this section we show that by using local optimizations we can move from the current feasible point to a point which reduces the potential by at least a constant. Let $a=(a_1,a_2,\cdots,a_N)$ be the current strictly interior feasible point. We first show that there is an ellipsoid strictly within the feasible region which contains a as well as a point which decreases the potential by a constant. We next show that by a series of minimizations of the convex function $f(z)$ over ellipsoids we can find such a point. In fact we can find a point which not only decreases the potential function by a constant but also reduces the objective function value. (This approach extends in a straight forward manner to reduce the problem of optimizing arbitrary convex functions over polytopes to a series of local optimizations; however the local optimizations appear difficult in most cases).

Consider the projective transformation $\hat{z}=T(z)=\frac{D^{-1}z}{e^TD^{-1}z}$, where $D=\mathrm{diag}[a_1,a_2,\cdots,a_N]$ and $z\in R^N$. This transformation maps a, where a satisfies $\sum_{i=1}^{N}z_i=M$ to $a_0=[1/N,\cdots,1/N]$, the center of the simplex $\{z:\ \sum_{i=1}^{N}\hat{z}_i=1,\ z\geq 0\}$. The inverse transformation is given by $z=T^{-1}(\hat{z})=M\left(\frac{D\hat{z}}{e^TD\hat{z}}\right)$ under the condition that $e^Tz=M$. Note that both the transformation and its inverse map straight lines to straight lines. Let S be the sphere in the transformed domain defined by

$$\{\hat{z}:\sum_{i=1}^{N}\hat{z}_i^{\,2}\leq\alpha^2/N(N-1)+1/N\}\cap\{\hat{z}:e^T\hat{z}=1\}$$

and let E be $T^{-1}(S)$. E contains the point a. Now as E is bounded, and can be expressed as the intersection of a convex region defined by a quadratic inequality and the hyperplane $e^Tz=M$, it is an ellipsoidal region. Moreover, its intersection with $e^TD^{-1}z=c$, where c is some constant, is also an ellipsoidal region. The feasible region for the optimization problem is the intersection of the affine space $\Omega=\{z:Ax+y=b\}$, the hyperplane $e^Tz=M$, and the positive orthant $z\geq 0$ where $z^T=[z_1,z_2,x^T,y^T]$. Since E is contained in the positive orthant, $E\cap\Omega$ is an ellipsoidal region that lies within the feasible region and contains a.

Let z_0 be the point where $f(z)$ is optimized in Ω. If z_0 is in $E\cap\Omega$ then optimizing $f(z)$ over the ellipsoidal region suffices to find the optimum. So suppose z_0 is not in $E\cap\Omega$. Let b be the point where the straight line joining a to z_0 intersects the boundary of $E\cap\Omega$. The next lemma shows that the potential decrease $P(a)-P(b)$ is greater than some constant. We first define the following useful variant of the potential function

$$\hat{P}(z)=N\ln(f(z)-f_0)-N\ln(e^TD^{-1}z)$$

w.r.t the current point a.

Lemma 1. Let $f(z)$ be a convex function. Then $\hat{P}(a)-\hat{P}(b)\geq\alpha$ and thus $P(a)-P(b)\geq\alpha-\frac{\alpha^2}{2(1-\alpha)}$.

Proof. Since b is the point where the straight line joining a to z_0 intersects the boundary of $E\cap\Omega$. $b=(1-\lambda)a+\lambda z_0$ and

$$T(b)=(1-\hat{\lambda})T(a)+\hat{\lambda}T(z_0)$$

where $\hat{\lambda}=\alpha/N$. As T maps straight lines to straight lines

$$(1-\hat{\lambda})=\frac{e^TD^{-1}a}{e^TD^{-1}b}(1-\lambda)$$

and since

$$f(b)-f_0\leq(1-\lambda)(f(a)-f_0)$$

$$f(b)-f_0\leq\frac{e^TD^{-1}b}{e^TD^{-1}a}(1-\alpha/N)(f(a)-f_0).$$

Thus $\hat{P}(a)-\hat{P}(b)\geq\alpha$. Also

$$P(a)-P(b)=N\ln\left(\frac{f(a)-f_0}{f(b)-f_0}\right)-\sum_{i=1}^{N}\ln\left(\frac{a_i}{b_i}\right)$$

$$\geq N\ln(1/(1-\alpha/N))-\sum_{i=1}^{N}\ln\left(\frac{\hat{a}_i}{\hat{b}_i}\right)$$

$$\geq\alpha-\frac{\alpha^2}{2(1-\alpha)}[4,sect.4,lemma\,4.2]\quad\blacksquare$$

We next show how to find a point which gives a constant reduction in potential. Consider the hyperplanes H_L and H_R defined by the equations

$$e^TD^{-1}z=c_L\quad\text{and}\quad e^TD^{-1}z=c_L(1+1/N^2)$$

respectively, where $c_L\leq e^TD^{-1}b\leq c_L(1+1/N^2)$. Let b_L and b_R be the points which minimize $f(z)$ over $E\cap\Omega\cap H_L$ and $E\cap\Omega\cap H_R$ respectively. And let b_E be the point which minimizes $f(z)$ over $E\cap\Omega$.

Lemma 2. One of the points b_E,b_L,b_R achieves a reduction of $\alpha-N\ln(1+1/N^2)$ in $\hat{P}(z)$ and thus a reduction of $\alpha-\frac{\alpha^2}{2(1-\alpha)}-N\ln(1+1/N^2)$ in $P(z)$.

Proof. First suppose that $c_L\leq e^TD^{-1}b_E\leq c_L(1+1/N^2)$. Then

$$f(b_E)-f_0\leq f(b)-f_0\leq\frac{e^TD^{-1}b}{e^TD^{-1}a}(1-\alpha/N)(f(a)-f_0)$$

and

$$\frac{e^TD^{-1}b}{1+1/N^2}\leq e^TD^{-1}b_E\leq e^TD^{-1}b(1+1/N^2)$$

Thus

$$\hat{P}(a)-\hat{P}(b_E) = N\ln(\frac{f(a)-f_0}{f(b_E)-f_0})-N\ln(\frac{e^TD^{-1}a}{e^TD^{-1}b_E})$$

$$\geq -N\ln(1-\alpha/N)-N\ln(\frac{e^TD^{-1}b}{e^TD^{-1}b_E})$$

$$\geq \alpha-N\ln(1+1/N^2)$$

Next suppose that $e^TD^{-1}b_E<c_L$ (the case where $e^TD^{-1}b_E>c_L(1+1/N^2)$ is similar). Since $f(z)$ is a convex function and as the straight line joining b_E to b intersects the plane $e^TD^{-1}z=c_L$, $f(b_L)<f(b)$. Thus

$$f(b_L)-f_0 \leq \frac{e^TD^{-1}b}{e^TD^{-1}a}(1-\alpha/N)(f(a)-f_0).$$

Also $\frac{e^TD^{-1}b}{(1+1/N^2)} \leq e^TD^{-1}b_L \leq e^TD^{-1}b$ and thus, as in the previous case,

$$\hat{P}(b_L)-\hat{P}(a) \geq \alpha-N\ln(1+1/N^2) \quad \blacksquare$$

Lemma 2 enables us to find a point which achieves the reduction of potential as follows: Consider the sequence of planes H_j defined as

$$H_j=\{z:\ e^TD^{-1}z=c_j,\ c_j=(1+j/N^2)e^TD^{-1}a\}\ ,\quad j=\cdots,-2,-1,0,1,2,\cdots$$

Let $H=\{\ H_j : H_j\cap E\cap\Omega\neq\phi\ \}$. Since the image of E is the sphere S with radius α/N, $N/(1+\alpha) \leq e^TD^{-1}z \leq N/(1-\alpha)$ and hence the ratio of the maximum to the minimum value of $e^TD^{-1}z$ in $E\cap\Omega$ is bounded by $(1+\alpha)/(1-\alpha)$. Moreover $e^TD^{-1}a=N$. There are thus $O(N^2)$ hyperplanes in the set H. Let b_j be the point that minimizes $f(z)$ over $E\cap\Omega\cap H_j$ and let b_E be the point that minimizes $f(z)$ over $E\cap\Omega$. By definition $c_{j+1} \leq c_j(1+1/N^2)$ and thus Lemma 2 allows us to conclude that one of the points in the set $\{b_E\}\cup\{b_j:H_j\cap E\cap\Omega\neq\phi\}$, gives the desired reduction in potential.

However, a point where the potential function is reduced by a constant may be found faster by using a form of binary search. First we need the following definitions and lemma. Let $P^*(j)=\min_{E\cap\Omega\cap H_j}\{\ln(f(z)-f_0)-\ln c_j\}$ where $H_j\in H$ is defined by $e^TD^{-1}z=c_j$. Let $P^*(j_0) = \min_{H_j\in H}P^*(j)$.

Lemma 3. Suppose $P^*(j)>P^*(j_0)$ and k lies in between j and j_0. Then

$$P^*(j) \geq P^*(k) + \frac{1}{2}\frac{j-k}{j-j_0}\frac{(1-\alpha)^2}{(1+\alpha)}(1-e^{P^*(j_0)-P^*(j)})$$

Proof. Let λ be such that $c_k=\lambda c_j+(1-\lambda)c_{j_0}$. Thus $\lambda=\frac{k-j_0}{j-j_0}$ and by convexity of $f(z)$ $f(b_k) \leq \lambda f(b_j)+(1-\lambda)f(b_{j_0})$. Thus

$$P^*(k) \leq \ln(\lambda(f(b_j)-f_0)+(1-\lambda)(f(b_{j_0})-f_0)) - \ln(\lambda c_j+(1-\lambda)c_{j_0}).$$

Now since

$$P^*(j_0)-P^*(j) = \ln(\frac{f(b_{j_0}-f_0)}{f(b_j-f_0)}) - \ln(\frac{c_{j_0}}{c_j})\ ,$$

$$\frac{f(b_{j_0})-f_0}{f(b_j)-f_0} = \frac{c_{j_0}}{c_j}e^{P^*(j_0)-P^*(j)}$$

and hence

$$P^*(k) \leq P^*(j) + \ln(1+\frac{1-\lambda}{\lambda}\frac{c_{j_0}}{c_j}e^{P^*(j_0)-P^*(j)}) - \ln(1+\frac{1-\lambda}{\lambda}\frac{c_{j_0}}{c_j}).$$

Finally,

$$P^*(j)-P^*(k) \geq -\ln(1-\frac{\frac{1-\lambda}{\lambda}\frac{c_{j_0}}{c_j}(1-e^{P^*(j_0)-P^*(j)})}{1+\frac{1-\lambda}{\lambda}\frac{c_{j_0}}{c_j}})$$

and since $\frac{(1-\alpha)}{(1+\alpha)} \leq \frac{c_{j_0}}{c_j} \leq \frac{(1+\alpha)}{(1-\alpha)}$ and $1-\lambda=\frac{j-k}{j-j_0}$ the result follows. ■

Lemma 3 is next used to show that only $O(\log N)$ optimizations of $f(z)$ need to be done.

Lemma 4. One of the points in the set $H^B=\{\ b_i : i=0,\pm 2^k$ where $0\leq k\leq\frac{\lceil\log(\ |H|\ )\rceil}{2}\}\cup\{b_E\}$ reduces the potential by at least $\frac{1}{8}\frac{((1-\alpha))^2}{(1+\alpha)}\alpha - \frac{\alpha^2}{2(1-\alpha)} - \frac{(1-\alpha)^2\alpha^2}{32(1+\alpha)N}$.

Proof. If b_E gives the desired reduction in potential then we are done. Thus suppose otherwise. By Lemma 2, $\hat{P}(a)/N - P^*(j_0) \geq \alpha/N$. Let $2^k\leq j_0<2^{k+1}$. Then by Lemma 3

$$P^*(2^k) \leq P^*(0) - \frac{1}{4}\frac{(1-\alpha)^2}{(1+\alpha)}(1-e^{P^*(j_0)-P^*(0)}).$$

Also let

$$P^*(0) = \hat{P}(a)/N - \gamma\ ,\quad \gamma>0$$

Now if $\gamma>\alpha/2N$ then since $P^*(j_0) \leq P^*(0)$, b_{2^k} gives the desired potential change. Otherwise

$$P^*(2^k) \leq \hat{P}(a)/N - \frac{1}{8N}\frac{(1-\alpha)^2}{(1+\alpha)}\alpha - \frac{(1-\alpha)^2\alpha^2}{32(1+\alpha)N^2}$$

Thus

$$\hat{P}(a)-\hat{P}(b_{2^k}) \geq \frac{1}{8}\frac{(1-\alpha)^2}{(1+\alpha)}\alpha - \frac{(1-\alpha)^2\alpha^2}{32(1+\alpha)N}$$

and the result follows. ■

In fact a point which not only reduces the potential by a constant but also reduces the objective function can be found by searching the set H^B. Redefine H_{j_0} to be the hyperplane on which there is a point which reduces the potential by a constant and also reduces the objective function. The existence of such a point is assured by Lemma 2. Now by the convexity of $f(z)$, $f(a) \geq f(b_{2^k}) \geq f(b_{j_0})$ in the proof of Lemma 4 above, and thus b_{2^k} gives a reduction in potential as well as a reduction in objective function. Alternatively, by using Lemma 3 a binary search may be applied on the set H to yield the desired point.

The algorithm we describe later will optimize over slightly shrunk and deformed ellipsoids and we next show that we are still assured of a sufficient reduction in potential. Suppose the ellipsoid E is shrunk by at most a factor of $(1+1/N^3)$ giving the ellipsoid E'. Lemma 1 still holds with point b redefined to be on the ellipsoid $E'\cap\Omega$ since the image of E' under the transformation $T(z)$ contains the sphere,

$$\{\hat z : \sum_{i=1}^{N} \hat z_i^2 \le (1+\alpha/N^3)^2 \alpha^2/N(N-1) + 1/N\} \cap \{\hat z : e^T \hat z = 1\},$$

where c is a constant. Let b_E be the point which optimizes $f(z)$ over $E' \cap \Omega$. And let b_L and b_R be the points which minimize $f(z)$ over $E_L \cap \Omega \cap H_L$ and $E_R \cap \Omega \cap H_R$ respectively, where H_L and H_R are defined as before in terms of the redefined b, and $E_L \supseteq E'$ and $E_R \supseteq E'$. With this redefinition, Lemma 2 remains valid. Finally, let b_j be the point which minimizes $f(z)$ over $E_j \cap \Omega \cap H_j$ where H_j are the hyperplanes defined as before and $E_j \supseteq E'$. Lemma 4 still holds and a point assuring a constant reduction in potential can be found as previously described.

1.3. Algorithm

We now present an algorithm to minimize a convex function $f(z)$ over a polytope. In order to measure potential changes the minimum value f_0 of the function $f(z)$ over the polytope is required. Since f_0 is unknown, we make use of a sliding objective function method. We maintain two parameters *HIGH* and *LOW* which serve as upper and lower bounds on f_0. The algorithm proceeds in stages. At the *jth* stage we have a guess g_j for f_0 such that $LOW < g_j < HIGH$, and during the *jth* stage we measure potential w.r.t. g_j. At each iteration during a stage we try to find a point that reduces the potential by a constant, by performing $O(\log N)$ local minimizations of $f(z)$ over ellipsoids contained within the polytope. The local minimizations are described by Problem 1 and Problem 2 which are described in Section 1.4. If one of the points obtained by these local minimizations reduces the potential as required then we proceed to the next iteration; otherwise the guess g_j is guaranteed to be less than f_0 [4], and we reset *LOW* to g_j. Once the objective function falls below a threshold u, $LOW < u < HIGH$, we reset *HIGH* to the current value of $f(z)$. A new stage starts whenever *LOW* or *HIGH* are reset, and a new guess for f_0 is then computed.

In $O(L)$ stages the difference $HIGH - LOW$ falls to $2^{-\Theta(L)}$; the total number of iterations in these $O(L)$ stages is $O(NL)$. Then we keep *HIGH*, *LOW*, and the guess for f_0, fixed, and in $O(NL)$ extra iterations we obtain a point where the value of $f(z)$ is at most $2^{-\Theta(L)}$ away from f_0. A proof of this is given in the Appendix. We then find an exact optimum as described in Section 1.7.

ALGORITHM QP
Begin
Let $z^0 = [z_1^0, z_2^0, x^0, y^0]$ be an initial point such that
  $Az^0 = b$, $Ax^0 < b$, $z^0 > 0$.
Let $LOW = 2^{-\Theta(L)}$, $HIGH = 2^{O(L)}$ and $\epsilon = 2^{-\Theta(L)}$.
$j := 0$; $t := 0$;
/* j and t are the stage and iteration numbers*/
While $HIGH - LOW \ge \epsilon$ do
  Begin
    $u := LOW + c_2(HIGH - LOW)$
    $g_j := LOW + c_1(HIGH - LOW)$
    /* c_2 and c_1 are appropriately chosen constants. */
    $P_j(z) := \sum_{i=1}^{N} \ln\left(\frac{f(z) - g_j}{z_i}\right)$
    While $f(z) > u$ and there is a constant reduction in potential $P_j(z)$ do
      Begin
      $D = \mathrm{diag}(z_1^t, \cdots, z_N^t)$
      Let b_E be the point that minimizes $f(z)$
        over the region defined by
        $Ax + y = b$
        $e^T z = M$
        $z^T D^{-2} z \le \left(\frac{(\alpha^2+N)^{1/2}}{N}(1+1/2N^3) e^T D^{-1} z\right)^2,$
      Let b_i, $i = 0, \pm 2^k$ where $0 \le k \le \frac{\lceil \log(\,|H|\,) \rceil + 1}{2}$, be
        the point that minimizes $f(z)$
        over the region defined by
        $Ax + y = b$
        $e^T z = M$
        $e^T D^{-1} z = e^T D^{-1} a (1 + i/N^2)$
        $z^T D^{-2} z \le \left(\frac{(\alpha^2+N)^{1/2}}{N}(1+1/2N^3) e^T D^{-1} z\right)^2$
      If one of the points b_E, b_i, $i = 0, \pm 2^k$
          where $0 \le k \le \frac{\lceil \log(\,|H|\,) \rceil + 1}{2}$,
        gives a constant reduction in potential then
          let z^{t+1} be one such point
        else
          let z^{t+1} be z^t
      $t := t + 1$
      end
    if $f(z) \le u$ then $HIGH := f(z)$
      else $LOW := g_j$
    $j := j + 1$
  end
If $f(z) - HIGH > \epsilon$ then
  let $g_j = HIGH$ and find a sequence of points with
  decreasing potential $P_j(z)$ until $f(z) - HIGH \le \epsilon$.
end.

1.4. Solving the local optimization problems

The algorithm described in the previous section solves the convex Quadratic programming problem, in $O(NL)$ iterations where each iteration comprises a minimization of the convex quadratic function $f(z)$ over an ellipsoid as described by Problem P1 and $O(\log N)$ similar minimizations as described by Problem P2. Problems P1 and P2 are described below.

Problem P1 $\quad \min f(x) = \frac{1}{2} x^T B x + p^T x$

$$\text{s.t. } Ax + y = b$$
$$e^T z = M$$
$$z^T D^{-2} z \le (r^1 e^T D^{-1} z)^2$$

where $z^T = [z_1, z_2, x^T, y^T]$, $N = n + m + 2$, $z \in R^N$, $x \in R^n$, $y \in R^m$, $A \in R^{m \times n}$, $b \in R^m$ $e^T = [1, 1, \ldots, 1] \in R^N$, and $r^1 = \frac{(\alpha^2+N)^{1/2}}{N}(1+1/2N^3)$, $D = \mathrm{diag}[a_1, a_2, \cdots, a_N]$ where $a = [a_{z_1}, a_{z_2}, a_3, \ldots, a_N]$ is the current point.

Problem P2 $\quad \min f(x) = \frac{1}{2} x^T B x + p^T x$

$$\text{s.t. } Ax + y = b$$
$$e^T z = M$$
$$H_j:\ e^T D^{-1} z = c_j = N(1 + j/N^2)$$
$$z^T D^{-2} z \le \left(\frac{(\alpha^2+N)^{1/2}}{N} e^T D^{-1} z\right)^2$$

where z, N, x, y, A, b and e^T are as defined above and $j = \pm 2^k$ where $0 \le k \le \frac{\lceil \log(\,|H|\,) \rceil}{2} + 1$. Each of the above problems is next reduced to the problem

Problem 2. $\quad \min \frac{1}{2} x^T B x + p^T x$

$$\text{s.t. } (x - x_G)^T G (x - x_G) \le 2r, \quad x \in R^n.$$

by elimination of the slacks y,z_1,z_2. We will then show how to solve Problem 2 efficiently.

Let $D = \text{diag}[a_{z_1},a_{z_2},D_x,D_y]$ where D_x and D_y denote the diagonal submatrices in D corresponding to the x and y coordinates respectively. We let a_x and a_y denote the vectors defined by $a_x^T = e_x^T D_x^{-1}$ and $a_y^T = e_y^T D_y^{-1}$ where $e_x^T = [1,1,...,1] \in R^n$ and $e_y^T = [1,1,\cdots,1] \in R^m$. To reduce Problem P1 first combine slacks z_1,z_2 into z_3 as $z_3=z_1+z_2$. On substituting $y_{P1}=b_{P1} - A_{P1}x$ where

$$y_{P1} = \begin{bmatrix} y \\ z_3 \end{bmatrix}, \qquad b_{P1} = \begin{bmatrix} b \\ M - e_y^T b \end{bmatrix}$$

$$A_{P1} = \begin{bmatrix} A \\ e_A^T \end{bmatrix}, \qquad e_A^T = e_x^T - e_y^T A$$

Problem P1 reduces to

$$\text{Problem2.} \quad \min \frac{1}{2} x^T B x + p^T x$$
$$\text{s.t.} \quad (x - x_{G_1})^T G_1 (x - x_{G_1}) \le 2r_1, \ x \in R^n.$$

where $G_1 = G_{11} + G_{12}$ and

$$G_{11} = D_x^{-2} + A^T D_y^{-2} A$$

$$G_{12} = (a_{z_1} + a_{z_2})^{-2} e_A e_A^T - (r^1)^2 (a_x a_x^T + A_{P1}^T a_{y_{P1}} a_{y_{P1}}{}^T A_{P1})$$
$$+ (a_x a_{y_{P1}}^T A_{P1} + A_{P1}^T a_{y_{P1}} a_x^T)$$

where $a_{y_{P1}}{}^T = [a_y^T, 1/(a_{z_1} + a_{z_2})]$ and

$$x_{G_1}^T = [(r^1)^2 (b_{P1}^T a_{y_{P1}} a_{y_{P1}}^T A_{P1} - b_{P1}^T a_{y_{P1}} a_x^T) - b_{P1}^T D_{y_{P1}}^{-2} A_{P1}] G_1^{-1}$$

$$2r_1 = x_{G_1}^T G_1 x_{G_1} - b_{P1}^T D_{y_{P1}}^{-2} b_{P1} + (r^1)^2 b_{P1}^T a_{y_{P1}} a_{y_{P1}}^T b_{P1}$$

Problem P2 is also reduced to the same form as Problem 2 by substituting $y_{P2} = b_{P2} - A_{P2}x$ where

$$y_{P2} = \begin{bmatrix} y \\ z_1 \\ z_2 \end{bmatrix}, \qquad A_{P2} = \begin{bmatrix} A \\ A_z^{-1} \begin{bmatrix} e_A^T \\ e_D^T \end{bmatrix} \end{bmatrix}$$

$$A_z = \begin{bmatrix} 1 & 1 \\ 1/a_{z_1} & 1/a_{z_2} \end{bmatrix}, \qquad e_A = e_x^T - e_y^T A,$$

$$e_D = e_x^T D_x^{-1} - e_y^T D_y^{-1} A$$

$$b_{P2} = \begin{bmatrix} b \\ A_z^{-1} \begin{bmatrix} M - e_y^T b \\ c_j - e_y^T D_y^{-1} b \end{bmatrix} \end{bmatrix},$$

Problem P2 reduces to

$$\text{Problem2.} \quad \min x^T B x + p^T x$$
$$\text{s.t.} \quad (x - x_{G_2})^T G_2 (x - x_{G_2}) \le 2r_2, \ x \in R^n,$$

where $G_2 = G_{21} + G_{22}$ and

$$G_{21} = D_x^{-2} + A^T D_y^{-2} A$$

$$G_{22} = [-e_A \ e_D] (A_z^{-1})^T \begin{bmatrix} a_{z_1}^{-2} & 0 \\ 0 & a_{z_2}^{-2} \end{bmatrix} A_z^{-1} \begin{bmatrix} -e_A^T \\ e_D^T \end{bmatrix}$$

$$x_{G_2} = G_2^{-1} (A_{P2}^T D_{yz}^{-1} b_{P2})$$

$$2r_2 = c_i^2 \frac{(\alpha^2 + N)}{N^2} - b_{P2}^T D_{yz}^{-2} b_{P2}$$

$$D_{yz} = \begin{bmatrix} D_y & & \\ & a_{z_1} & \\ & & a_{z_2} \end{bmatrix}$$

1.5. Optimizing Convex Quadratic functions over Ellipsoids

We next show how to efficiently solve the problem

$$\text{Problem 2.} \quad \min x^T B x + p^T x$$
$$s.t. \quad (x - x_G)^T G (x - x_G) \le 2r, \ x \in R^n$$

By application of a linear transformation the above problem may be converted to the following form;

$$\text{Problem 3.} \quad \min x_1^T x_1 + Q^T x_2$$
$$s.t. \quad (x - x_0)^T \Lambda (x - x_0) \le 2r$$

where Λ is a diagonal matrix with positive entries and $x^T = [x_1^T, x_2^T]$, $x_1 \in R^k$, $x_2 \in R^{n-k}$. We assume that the optimum to Problem 2 lies on the boundary of the ellipsoid $(x - x_G)^T G (x - x_G) \le 2r$, $x \in R^n$; otherwise B is necessarily non-singular and the optimum is given by $x = \frac{-B^{-1}p}{2}$.

To solve Problem 2 we first characterize the solution to Problem 3. By the theory of Lagrange multipliers the optima to Problem 3 are given by the solution to the set of equations

$$\begin{bmatrix} x_1 \\ q \end{bmatrix} = \mu \Lambda (x - x_0) \qquad (1)$$

$$(x - x_0)^T \Lambda (x - x_0) = 2r \qquad (2)$$

Let $\Lambda = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$. This gives

$$x_i = \mu \lambda_i (x_i - x_{0_i}) \quad i = 1, \cdots, k$$

$$q_j = \mu \lambda_j (x_j - x_{0_j}) \quad j = k+1, \cdots, n.$$

Substituting into the ellipsoid equation (2), gives

$$\Psi(\mu) = \frac{1}{2} \sum_{i=1}^{k} \frac{x_{0_i}^2 \lambda_i}{(1 - \mu \lambda_i)^2} + \frac{1}{2} \sum_{j=k+1}^{n} \frac{q_j^2}{\mu^2 \lambda_j} = r.$$

Firstly note that when x minimizes the objective function μ must be negative. Suppose otherwise, i.e. that μ is positive. Then x_i and $x_i - x_{0_i}$ have the same sign for all i and thus decreasing x_i decreases the objective function while staying within the ellipsoid defined in (2). Moreover $\Psi(\mu)$ is a monotonically increasing function of μ, for $\mu < 0$. Thus a solution to $\Psi(\mu) = r$ can be obtained by a binary search on μ, $\mu < 0$. Let $\Psi(\mu^*) = r$, for $\mu^* < 0$. We next show that it suffices to find a μ' s.t. $\mu^*(1 + 1/2n^3) \le \mu' \le \mu^*$. This choice of μ ensures that $r/(1 + 1/2n^3) \le \Psi(\mu') \le r$, and the corresponding x optimizes the objective function over the slightly shrunk ellisoid, $(x - x_0)^T \Lambda (x - x_0) \le 2\Psi(\mu')$. This corresponds to optimizing over the slightly shrunk ellipsoid $(x - x_0)^T G (x - x_0) \le 2\Psi(\mu')$ in Problem 2.

Now, as shown in section 3, when both Problems $P1$ and $P2$ are reduced to problems like Problem 2, G is obtained by intersecting the ellipsoid E and then projecting onto a subspace. Suppose E is shrunk by some factor γ. Then $E\bigcap\Omega$, where Ω is an affine space, is shrunk by at least γ', $\gamma'>\gamma$, and the ellipsoid which is the projection of $E\bigcap\Omega$ onto some subspace is shrunk by the same factor γ'. Thus optimizing over $(x-x_0)^T G(x-x_0) \le 2\Psi(\mu')$ corresponds to optimizing over $E'\bigcap\Omega$ in Problem P1 and over $E_i'\bigcap\Omega\bigcap H_i$ in problem P2, where E' is obtained by shrinking E by a factor of at most $(1+1/2n^3)$. As described in section 1.2 these optimizations suffice to find a sufficient reduction in potential.

We next note that the value of the Lagrange multiplier μ remains invariant under linear transformations since the corresponding equations characterizing the multiplier μ are equivalent. As Problem 2 and Problem 3 are related by a linear transformation, it suffices to find a μ' such that $\mu^*(1+1/2n^3) \le \mu' \le \mu^*$, where μ^* satisfies

$$Bx + p = \mu G(x-x_G)$$
$$(x-x_G)^T G(x-x_G) = 2r, \quad x \in R^n.$$

In our algorithm the value of μ^* is upper and lower bounded by $2^{O(L)}$ and $2^{-O(L)}$ respectively. So to obtain the desired approximation μ' to μ^* in Problem 2 we simply solve the system of equations

$$2Bx + p = 2\mu G(x-x_G)$$

for $O(\log L)$ values of μ. The complexity of this procedure is $O(n^3(\log L))$ arithmetic operations, using standard linear equation solvers.

Using this solution to Problem 2, the algorithm for Quadratic programming requires $O(n^3 N(\log N)(\log L)L)$ arithmetic operations.

1.6. Improving the complexity

In this section we show how to reduce the complexity of the Quadratic Programming problem by solving slightly modified versions of problems P1 and P2 during each iteration. The modified problems differ in that the quadratic form $z^T D^{-2} z$ changes to $z^T D_\Delta^{-2} z$ in both the problems where $D_\Delta = D\Delta$ and $\Delta = diag[\Delta_1, \cdots, \Delta_N]$, $\Delta_i^2 \in [1/2, 2]$. and α is replaced by $\alpha/2$. The quadratic constraint now defines a slightly twisted and shrunk ellipsoid. By the results of section 1.2 a constant reduction in potential can still be found. Let $D_\Delta^i = \text{diag}[d_{\Delta_1}^i, \cdots, d_{\Delta_N}^i]$ and $D^i = \text{diag}[z_1^i z_2^i, \cdots, z_N^i]$ be the matrices D_Δ and D at the ith iteration. ($z^i = [z_1^i, \cdots, z_N^i]$ is the point at the beginning of the ith iteration.) Moreover let $\sigma_i = \frac{1}{N}\sum_{j=1}^{N}\frac{z_j^{i+1}}{z_j^i}$. Initially $D_\Delta^0 = D^0$ and D_Δ^{i+1} is obtained from D_Δ^i as follows

If $\sigma_i d_{\Delta_j}^i \in [\frac{z_j^{i+1}}{\sqrt{2}}, \sqrt{2} z_j^{i+1}]$ then
let $d_{\Delta_j}^{i+1} = \sigma_i d_{\Delta_j}^i$ (Type 1 Change)
else $d_{\Delta_j}^{i+1} = \sigma_i d_{\Delta_j}^i + (z_j^{i+1} - \sigma_i d_{\Delta_j}^i)$ (Type 2 Change)

The total number of such changes has been shown to be bounded by $O(N^{1.5}L)$ throughout the execution of the algorithm in [4]. We use this fact to improve the time complexity of the Quadratic Programming algorithm. In our algorithm we express D_Δ^i as $D_H + D_L$, where D_H is a high rank matrix and $D_L = \text{diag}[d_{L_1}, \cdots, d_{L_N}]$ is a low rank matrix which accumulates the type 2 changes. Initially $D_H = D^0$ and $D_L = 0$. At the end of the ith iteration D_H and D_L are updated as follows:

Procedure Update D_H, D_L
$D_H = \sigma_i D_H$
$D_L = \sigma_i D_L$
If $\sigma_i d_{\Delta_j}^i \notin [\frac{z_j^{i+1}}{\sqrt{2}}, \sqrt{2} z_j^{i+1}]$
then $d_{L_j} = d_{L_j} + (z_j^{i+1} - \sigma_i d_{\Delta_j}^i)$

Now in order to solve the modified problems $P1$ and $P2$ we proceed as in the previous section and reduce each of the problems to Problem 2. The matrices in Problem 2 corresponding to Problem P1 and P2 are $G_1 = G_{11} + G_{12}$ and $G_2 = G_{12} + G_{22}$ respectively. Note that $G_{11} = G_{21}$ and G_{12} and G_{22} are constant rank matrices. Let G_{11}^i denote the matrix G_{11} in Problem 2 at the ith iteration. The decomposition $D_\Delta^i = D_H + D_L$ induces a decomposition of G_{11}^i into $G_{11}^i = G_H + G_L$. where

$$G_H = D_{Hx}^{-2} + A^T D_{Hy}^{-2} A,$$

$$G_L = D_{Lx}^{-2} + A^T D_{Ly}^{-2} A$$

and D_{Hx}, D_{Hy} are the entries in D_H corresponding to the x and y coordinates respectively. Similarly D_{Lx}, D_{Ly} are the entries in D_L corresponding to the x and y coordinates respectively. Now in the ith iteration G_H changes by a scale factor only and each Type 2 change in D_L induces a rank one change in G_L. We will show that as long as G_L has low rank and some matrices related to G_H have been precomputed, Problem 2 (and hence Problem P1 and Problems P2) can solved quicker than before. Unfortunately this procedure is not economical when the rank of G_L exceeds a certain threshold. At this stage we reset D_H, G_H, D_L, G_L and recompute the required matrices. A value for this threshold (which is $=N^{2.5/3}$) is obtained by balancing the number of operations reqired to recompute the required matrices and the number of operations required to solve Problem $P1$ and $O(\log N)$ Problems $P2$.

We need to precompute (using $O(n^3)$ operations) the following matrices related to G_H

(1) The Cholesky factorization $G_H = LL^T$ and L^{-1} where L is a lower triangular matrix.

(2) The tridiagonaliztion $(L^{-1})B(L^{-1})^T = QTQ^T$, where Q is a unitary matrix and T is a tridiagonal matrix.

(3) The products $(L^{-1})^T Q$ and $(L^{-1})^T QA$.

We next give a modified algorithm assuming that the optimum value of the objective function is zero; the sliding objective function method may be incorporated in a manner similar to that in the algorithm given in section 1.3.

Modified Algorithm

Let z^0 be an initial point such that
$Az^0 = b$, $Ax^0 < b$, $z^0 > 0$
$D_H := D_\Delta^0 = D^0$
$D_L := 0$
$G_H = G_{11}^0 = G_{21}^0$, $\epsilon = 2^{-\theta(L)}$,
Compute L, L^{-1}, Q, T, $(L^{-1})^T Q$, $(L^{-1})^T QA$
While $f(z) > \epsilon$ and there is a constant reduction in $\sum_{i=1}^{N}\ln(\frac{f(z)}{z_i})$ do

Begin (ith iteration)
Find a point that decreases the potential by a constant by solving Problem $P1$ and $O(\log N)$ Problems $P2$.
Each problem is solved by reducing it to problem 2 which is solved approximately using precomputed matrices.
Update D_H, D_L.
If (rank of D_L) > threshold ($= N^{2.5/3}$) then
$D_H := D^{i+1}$, $D_L := 0$
Compute L, L^{-1}, Q, T,
$(L^{-1})^T Q$, $(L^{-1})^T QA$

end

In order to find a point that assures a reduction in potential in the above algorithm we need to solve the following form of Problem 2,

$$\min \frac{1}{2} x^T Bx + p^T x$$
$$\text{s.t.} (x - x_G)^T G(x - x_G) \le r, \ x \in R^n.$$

where $G = G_H + G_L + G_q$. G_q is a matrix with constant rank and we are given the Cholesky factorization $G_H = LL^T$, the tridiagonalization $(L^{-1})B(L^{-1})^T = QTQ^T$ and the products $(L^{-1})^T Q, (L^{-1})^T QA$. Moreover G_L is a matrix of low rank and can be expressed as

$$G_L = D_{Lx}^{-2} + A^T D_{Ly}^{-2} A$$

where D_{Lx} and D_{Ly} are diagonal submatrices of D_L corresponding to the x and y coordinates. Each has at most $N^{2.5/3}$ entries. $G_L + G_q$ can thus be expressed as

$$G_L + G_q = U_1 V_1^T + A^T U_2 V_2^T A$$

where U_1, V_1, U_2, V_2 are matrices with n rows and $t = O(N^{2.5/3})$ columns. On applying the linear transformation $w = Q^T L^T x$, Problem 2 becomes

$$\min \ w^T Tw + p_w^T w$$
$$\text{s.t} \ (w - w_G)^T (I + U_3 V_3^T + U_4 V_4^T)(w - w_G) \le r$$

where $p_w^T = p^T (L^T)^{-1} Q$, $w_G = Q^T L^T x_G$ and

$$U_3 = Q^T L^{-1} A^T U_2, \quad V_3 = Q^T L^{-1} A^T V_2,$$
$$U_4 = Q^T L^{-1} U_1, \quad V_4 = Q^T L^{-1} V_1.$$

Note that having precomputed the products $(L^{-1})^T Q$ and $(L^{-1})^T QA$, U_3, V_3, U_4, V_4 can be computed in $O(nt)$ arithmetic operations.

A solution to the transformed problem is obtained, using the method outlined in section 1.5, by solving the following system of equations for $O(\log N + \log L)$ values of μ.

$$2Tw + p_w = -2\mu(I + U_3 V_3^T + U_4 V_4^T)(w - w_G)$$
$$= -2\mu(I + U_5 V_5^T)(w - w_G)$$

where $U_5^T = [U_3^T, U_4^T]$ and $V_5^T = [V_3^T, V_4^T]$. Let $T + \mu I = R_1$ and let $T + \mu I + \mu U_5 V_5^T = R_2$. The above system of equations can be now rewritten as

$$2R_2 w = -p_w + 2(R_2 - T) w_G$$

and is solved as follows,

(1) Compute R_1^{-1}. Since R_1 is tridiagonal this can be done with $O(n^2)$ arithmetic operations.

(2) Compute $R_3 = R_1^{-1} U_5$ and $R_4 = R_1^{-1} V_5$ by solving $R_1 w = U_5$ and $R_1 w = V_5$. This requires $O(nt)$ arithmetic operations.

(3) Express
$$R_2^{-1} = R_1^{-1} - \mu R_1^{-1} U_5 [I + \mu V_5^T R_1^{-1} U_5]^{-1} V_5^T R_1^{-1}$$
$$= R_1^{-1} - \mu R_3 R_5^{-1} R_4$$
In this expression $V_5^T R_1^{-1} U_5$ is computed in $O(nt^2)$ operations. Then computing R_5^{-1} requires an additional $O(t^3)$ operations. R_2^{-1} is not computed explicitly but left in the second form.

(4) Finally the solution w is obtained by postmultiplying the expression for R_2^{-1} by the vector $-\frac{1}{2} p_w + (R_2 - T) w_G$ in $O(n^2)$ operations.

The total number of arithmetic operations required in the entire algorithm are $O(n^3 N^{2/3} L + nN^{5/3} NL(\log N + \log L)) = O(N^{3.67}(\log N + \log L)L)$ since

(1) It requires $O(n^3)$ operations to compute L, Q, T and the associated products. These computations are performed $O(N^{1.5} L / N^{2.5/3})$ number of times.

(2) At each iteration $O(nt^2) = O(nN^{5/3})$ operations are required to solve each of the $O(\log N + \log L)$ modified Problems 2.

1.7. Finding an exact optimum

In this section we describe how to find an exact optimal solution once we have a solution that is very close in objective function value to the optimum. Note that there is a point with rational coordinates which minimizes the convex quadratic form over the polytope. Consider a maximal set of inequalities, say $A_1 x \le b_1$, that are satisfied with equality at an optimum point. Then every solution to the problem

$$\min \ \frac{1}{2} x^T Bx + p^T x$$
$$\text{s.t.} \ A_1 x = b_1$$

is a solution to the original problem. Using the theory of convex programming [10], the solutions to the above problem are characterized by the following system of equations,

$$Bx + p = A_1^T \lambda$$
$$A_1 x = b_1$$

the solution of which has the desired rational coordinates. (λ is the Lagrange multiplier.)

We first consider the case when the matrix B in the quadratic form is positive definite. Let x_{opt} be an optimum, let $x = x_{opt} + \Delta x_{opt}$ be a point in the polytope, and let $\frac{1}{2} x^T Bx + p^T x = \frac{1}{2} x_{opt}^T B x_{opt} + p^T x_{opt} + \Theta$. Since the polytope lies entirely to one side of the hyperplane which is tangential to the surface defining the objective function at the optimum point, $(Bx_{opt} + p)^T x \ge (Bx_{opt} + p)^T x_{opt}$ for every point x in the polytope. Thus it follows that $(\Delta x_{opt})^T B (\Delta x_{opt}) \le \Theta$, and

$$\|\Delta x_{opt}\|_2^2 \le \Theta / (\text{smallest eigenvalue of B})$$

Since the smallest eigenvalue of B is greater than $\frac{1}{n 2^L}$, choosing $\Theta \le 2^{-(5L + 2 + \log_2 n)}$ ensures that $\|\Delta x_{opt}\|_2 < 2^{-(2L+1)}$.

We find an x, a required approximation to the optimum, by letting $\epsilon = \Theta$ in the algorithm in section 1.3. The optimum point is found by using continued fractions to jump to the unique rational point, with denominators and numerators less than 2^L, closest to the point obtained after execution of the algorithm in section 1.3.

Next suppose that the matrix B in the quadratic form is positive semi-definite. In this case the optimum point is not unique but for any two optima x_{opt} and x_{opt}', $Bx_{opt} = Bx_{opt}'$. Since there exists an optimum point x_{opt} that has rational coordinates, the numerators and the common denominator being integers less than 2^L, Bx_{opt} and $p^T x_{opt}$ also have rational coordinates with the numerators and the common denominator being integers less than 2^{2L} as B and p have integer entries. Let $x = x_{opt} + \Delta x_{opt}$, where x_{opt} is an optimum, and let $\frac{1}{2}x^T Bx + p^T x = \frac{1}{2}x_{opt}^T Bx_{opt} + p^T x_{opt} + \Theta$. Also let $x_{opt} = x_{opt_1} + x_{opt_2}$, $x = x_1 + x_2$ and $\Delta x_{opt} = \Delta x_{opt_1} + \Delta x_{opt_2}$ where x_{opt_1}, x_1 and Δx_{opt_1} are in the row space of B whereas x_{opt_2}, x_2 and Δx_{opt_2} are in the null space of B. Then $\frac{1}{2}x_1^T Bx_1 + p^T x = \frac{1}{2}x_{opt_1}^T Bx_{opt_1} + p^T x_{opt} + \Theta$, and by an argument similar to that in the previous case,

$$\|\Delta x_{opt_1}\|_2^2 \leq \Theta/(\text{smallest non-zero eigenvalue of B })$$

An optimum point is found as follows. First find an x such that $\Theta < 2^{-(7L+3\log_2 N+2)}$, and compute Bx_{opt} by evaluating Bx and using the method of continued fractions to find Bx_{opt}. Similarly, compute $p^T x_{opt}$ from $p^T x$. An optimum point is then a solution to the following feasibility problem.

$$\begin{aligned} Bx &= Bx_{opt} \\ p^T x &= p^T x_{opt} \\ Ax &\leq b \\ x &\geq 0 \end{aligned}$$

1.8. Precision of Arithmetic Operations

In this section we show that it is adequate to perform arithmetic operations to a precision of $O(L)$ bits. We asume that the polytope defined by $Ax \leq b$, $x \geq 0$, is non-degenerate (if not we can work with the polytope $Ax \leq b + 2^{-O(L)}$, $x \geq 0$ which is guranteed to be non-degenerate [7]).

Initially, we start with a point in the interior of the slightly modified polytope

$$Ax \leq b - \beta^0, \; x \geq \beta^0, \; \beta^0 = 2^{-k_1 L}$$

At the beginning of the i^{th} iteration we have a point x^i and a slack vector y^i located in the polytope P^i defined by

$$Ax + y = b - \beta^i, \; x_j \geq 2^{-k_2 L}, \; y_j \geq 2^{-k_2 L}, \; 2^{-k_1 L} \geq \beta^i \geq 2^{-k_2 L}$$

During the i^{th} iteration we do local optimizations over ellipsoids contained in the polytope P^i and obtain a point $(x^i)'$ and a slack vector $(y^i)'$ which reduce the potential. $(x^i)'$ and $(y^i)'$ are still in the polytope P^i. The components of $(x^i)'$ and $(y^i)'$ are rounded off to multiples of $2^{-k_2 L}$ to obtain x^{i+1} and y^{i+1}. x^{i+1} together with the slack y^{i+1} may no longer lie in the polytope P^i. However, x^{i+1}, y^{i+1} satisfy $Ax + y \leq b - \beta^i + (n+1)(a_{ij})_{\max} 2^{-k_2 L}$, where $(a_{ij})_{\max}$ is the entry with the largest magnitude among all the entries in the constraint matrix A. So we can find β^{i+1} such that $0 \leq \beta^i - \beta^{i+1} \leq (n+1)(a_{ij})_{\max} 2^{-k_2 L}$.

The number of iterations is bounded by γNL for some constant γ, and we choose k_2 so that the final point at the end of the last iteration is within the original polytope $Ax \leq b$, $x \geq 0$. We choose k_1 so that the optimum value of the objective function over the modified polytope $Ax \leq b - \beta^0$, $x \geq \beta^0$, differs from the optimum value over the original polytope $Ax \leq b$, $x \geq 0$, by at most Θ^2 where $\Theta = 2^{-7L+\log_2 N+2}$. Once we have a point where the value of the objective function differs from the optimum value over the original polytope by at most Θ we can use the method described in Section 1.7 to find an exact optimum over the original polytope. So let us suppose that the value of the objective function at x^i differs from the optimum value over the original polytope by at least Θ. Then, by a series of local optimizations over ellipsoids contained in the polytope P^i, we can find a point $(x^i)'$, and a slack $(y^i)'$, which decrease the potential by a constant, and the rounding process changes the potential by a negligible amount.

We choose M in the equation $e^T z = M$ large enough so that the slacks z_1 and z_2 are always greater than or equal to $n^3 2^L$. Since the largest value of any co-ordinate of a feasible point x and slack y is bounded by 2^L, the relative changes in z_1 and z_2 are negligible, and so z_1 and z_2 have a negligible effect on the potential.

We shall now bound the condition numbers of the matrices describing the ellipsoids over which local optimizations are performed. During each stage we have to solve Problem P1 and Problem P2 as described in Section 1.4. Problem P1 reduces to

$$\begin{aligned} &\min \frac{1}{2}x^T Bx + p^T x \\ &\text{s.t.} \quad (x - x_{G_1})^T G_1 (x - x_{G_1}) \leq 2r_1, \; x \in R^n. \end{aligned}$$

where G_1 is the matrix describing the projection of $E \cap \Omega$, where $E = \{z : e^T z = M\}$, $\Omega = \{z : Ax + y = b\}$, and $z^T = [x^T, y^T, z_1, z_2]$, onto the space of the $x's$.

As the condition number of the matrix describing an ellipsoid does not increase on intersecting the ellipsoid with an affine space $\kappa(E \cap \Omega) \leq \kappa(E)$. Moreover,

$$\kappa(G_1) \leq \kappa(E \cap \Omega)(1 + \text{largest eigenvalue of } A^T A)$$

$\kappa(E)$ is bounded as follows: Let a be the current point,

$$\kappa(E) \leq \frac{\text{largest distance from } a \text{ to boundary of } E}{\text{smallest distance from } a \text{ to boundary of } E}$$

The largest distance from a to boundary of E is at most $n2^L$. Let b be the point closest to a on the boundary of E. Suppose $|e^T D^{-1} b - e^T D^{-1} a| \geq 1/n$, then $\|a - b\|_2 \geq 1/(n\|D^{-1}e\|_2) \geq 2^{-k_2 L}/n^2$; otherwise $\|a - b\|_2 \geq (\|D^{-1}e\|_2 / e^T D^{-1} a) \geq \alpha 2^{-k_2 L}/n^2$, since $T(a)$ is the center of the sphere, S, in the transformed domain, and b lies on the boundary of this sphere. Thus $\kappa(E) \leq n^3 2^{(k_2+1)L}$.

Problem P2 is also reduced to the same form as Problem 2. Problem P2 reduces to

$$\begin{aligned} &\min \frac{1}{2}x^T Bx + p^T x \\ &\text{s.t.} \quad (x - x_{G_2})^T G_2 (x - x_{G_2}) \leq 2r_2, \; x \in R^n. \end{aligned}$$

where $G_2 = G_{21} + G_{22} = D_x^{-2} + A^T D_y^{-2} A + G_{22}$, D_x, and D_y are as defined in Section 1.4. Since G_2 is the sum of positive semidefinite matrices, the smallest eigenvalue of $G_2 \geq$ smallest eigenvalue of D_x^{-2}. Hence

$$\kappa(G_2) \leq \|G_2\|_2 / (\text{ smallest entry of } D_x^{-2})$$

$$\leq O(2^{(14k_2+2)L}$$

Having bounded the condition number of the ellipsoid matrices, we next show that $O(L)$ precision is adequate to give a sufficient reduction in potential. As described in Section 1.4 at each iteration we solve the following problem

$$\min \frac{1}{2}x^T Bx + p^T x$$
$$\text{s.t.} \quad (x-x_G)^T(G_H+G_L+G_q)(x-x_G) \leq 2r, \quad x \in R^n.$$

First the Cholesky factorization $G_H=LL^T$ and L^{-1} are computed. Next $L^{-1}B(L^{-1})^T$ is tridiagonalized to T using unitary matrices whose product is Q. The above computations are performed using algorithms in [2, 8, 9] with k_3L precision. Let Q_c, T_c , L_c, and $(L^{-1})_c$, be the computed matrices. The computed Q and T are used to transform the above problem to

$$\min \frac{1}{2}w^T T_c w + p_{w_c}^T w$$
$$(w-w_{G_c})^T(I+U_c V_c+R_c)(w-w_{G_c}) \leq r$$

as in section where $p_{w_c} = p^T L^{-1} Q^T$. This problem is equivalent to

Problem P^* $\quad \min \frac{1}{2}w^T(T+E_T)w + (p_w+\Delta p_w)^T w$

$$(w-w_G+\Delta w_G)^T(I+UV+R+E_G)(w-w_G+\Delta w_G) \leq r$$

where E_G, E_T are small error matrices and Δp_w, Δw_G are small error vectors [8, 9]. Problem P^* is in turn equivalent to

$$\min \frac{1}{2}x^T(B+\Delta B)x + (p+\Delta p)^T x, \quad x \in R^n$$
$$\text{s.t.} \quad (x-x_G+\Delta x_G)^T(G_H+G_L+G_q+\Delta G)(x-x_G+\Delta x_G) \leq 2r$$

The 2–norms of the error matrices ΔB, ΔG, and the error vectors Δp, Δx_G, are bounded by $2^{(k_5-k_3)L}$ for some constant k_5. We briefly sketch how the bound is obtained [8, 9]. We have

$$\|\Delta B\|_2 \leq \|(L^{-1})_c^{-1}\|_2^2 \, \|E_T\|_2.$$

Since $\|(L^{-1})_c^{-1}\|_2^2 \leq 2\|G^1\|_2$, $\|E_T\| \leq O(n^2\|L^{-1}\|_2^2 \|B\|_2 2^{-k_3L})$, we get $\|\Delta B\|_2 \leq O(2^{(24k_2+8-k_3)L})$. And

$$\|\Delta p\|_2 \leq \|p\|_2 \, \|L^{-1}\|_2 \, 2^{-k_3L} \leq O(2^{(k_2+3-k_3)L})$$

Also

$$\|\Delta G\|_2 \leq O(\|E_G\|_2 \|L\|_2^2)$$
$$\leq O(2^{(26k_2+4-k_3)L})$$

since

$$\|E_G\|_2 \leq O(\|G_H\|_2 + \|G_L\|_2 + \|G_q\|_2) \, \|L_c^{-1}\|_2^2 \, 2^{-k_3L}$$
$$\leq O((\kappa(G_H) + \|G_L\|_2 + \|G_q\|_2))2^{-k_3L})$$

Solving the problem P^* gives a point $x_c = x+\Delta x$ where Δx is the error introduced in solving the system of equations $(T+\mu(I+U_c V_c+R_c))w=r$ and in transforming the point back to the original domain. Thus

$$\Delta x \leq n^2\|G\|_2^6 \, \|A\|_2^4 \, \mu^3 2^{-k_3L} + n^2\|G\|_2 \, \|\Delta Q\|_2$$
$$\leq O(2^{(76k_2+18+3k_4-k_3)L})$$

as $2^{k_4L} \geq \mu \geq 2^{-k_4L}$ and ΔQ is the error in each of the unitary matrices used in the tridiagonalization. The point x gives a sufficient reduction in potential since the ellipsoid corresponding to $G_H+G_L+G_q+\Delta G$ is a slightly twisted and shrunk version of the ellipsoid defined by $G_H+G_L+G_q$ for a suitable choice of k_3. Also Δx is of a much lower order than 2^{-k_2L} for $k_3 \geq 77k_2+20+4k_4$ and then the point x_c gives a sufficient reduction in potential.

2. Multicommodity Flows

2.1. Introduction

We consider the problem of finding a multicommodity flow in a directed network (V,E) [3]. The network has a set of sources S and sinks T, and it is required that source S_i send f_i units of commodity i to sink T_i through the network. Moreover, for each edge e_i there is a capacity c_i which upper bounds the total of all the commodities that may pass through that edge. For each of the sets V, E, S and T, we shall use the corresponding lower case letter to denote the size of the set.

For each source-sink pair (T_i, S_i) we add an edge directed from T_i to S_i, with an upper bound of f_i on the flow of commodity i through this edge. The goal is to find a flow such that the flow of commodity i in edge (T_i, S_i) equals the capacity f_i. No augmenting path algorithm is known for this problem. The multicommodity flow problem is formulated as the following linear program.

Problem MF.

$$\min \; p^T w = \sum_{i=1}^{s} y_i,$$
$$\text{s.t.} \quad APx = 0 \quad \text{... Flow Conservation}$$
$$Cx + Iy - zc = 0 \quad \text{... capacity constraints}$$
$$\sum_{i=1}^{n_1} x_i + \sum_{i=1}^{e} y_i + z = \sum_{i=1}^{e} c_i + 1$$
$$x \geq 0, \; y \geq 0, \; z \geq 0$$

where

$x \in R^{n_1}$ is the vector of flow variables. For each edge not incident on a source or sink vertex there are s variables each corresponding to one of the s commodities; for an edge incident on S_i or T_i there is exactly one variable corresponding to commodity i, and the first s coordinates of x correspond to the flows in the edges (T_i, S_i), $i=1,...,s$. $y \in R^e$ is the vector of slacks. $w^T=(x^T, y^T, z)$, and $w \in R^N$ where $N=n_1+e+1$. c is the capacity vector upper bounding x. Let n_2 be the number of edges incident either on source or sink vertices, and $n_3=e-n_2$. $C = \begin{pmatrix} I & 0 \\ 0 & C_1 \end{pmatrix}$, where $C \in R^{e \times n_1}$, $I \in R^{n_2 \times n_2}$, $C_1 \in R^{n_3 \times sn_3}$, and the i^{th} row of C_1 has 1's in the positions $s(i-1)+1$ through $s\,i$, and 0's in the remaining positions. A is a block diagonal matrix with the i^{th} block being the incidence matrix of the directed graph induced by the vertex set consisting of S_i and the vertices reachable from S_i, and P is an appropriate permutation matrix.

If a required flow does exist then the minimum value of $p^T w$ is zero, and a solution to Problem MF gives the required flow.

We adapt Karmarkar's linear programming algorithm [4] to give a procedure requiring $O(s^{3.5} v^{2.5} e L)$ arithmetic operations performed to a precision of $O(L)$ bits, where $L = \log(d_{\max}) + \log(\sum_i c_i) + \log N$. $d_{\max}$ is the largest absolute value of the determinant of any square submatrix of the constraint matrix in Problem MF. To get an initial strictly interior feasible point, we start with small positive flows of each commodity in each edge such that flow conservation and capacity constraints are satisfied, and the flow in an edge (T_i, S_i) of commodity i is less than f_i.

Applying Karmarkar's algorithm to Problem MF reduces the global problem to a sequence of $O(NL)$ local optimizations. Since the optimal value of the objective function is unknown, a sliding objective function method is employed without increasing the time complexity. In Section 2.2 we show how to reduce the cost of local optimizations by eliminating capacity constraints and inverting a matrix of lower dimension. In Section 2.3 we show how the amortized cost may be further reduced to give the desired bound on the total number of arithmetic operations. In Section 2.4 we describe how an optimum point may be obtained once we have a point where the objective function value is close to the optimum value. In Section 2.5 we show that it is adequate to perform all arithmetic operations in the algorithm to a precision of $O(L)$ bits. Interestingly enough, the minimum cost multicommodity flow problem can also be solved in the same time complexity. The approach outlined in Sections 2.2 and 2.3 extends in a straightforward manner to provide efficient solutions to similarly structured linear programs, for example, those arising in problems with generalized upper bounding and block angular problems [1].

2.2. Reducing cost of local optimizations

As described in [4], the algorithm generates a sequence of points $w^0, w^1, \ldots$ where w^0 is an initial feasible point as described before. w^{k+1} is obtained from w^k by a local optimization as follows.

(1) Find a direction w^d by solving the local optimization problem

$$\begin{aligned} \min\ & p^T D w \\ & APD_x x = 0 \\ & CD_x x + D_y y - cz = 0 \\ & e^T w = 0 \\ & w^T w \le 1 \end{aligned}$$

where $D = \mathrm{diag}(D_x, D_y, 1)$, $D_x = \mathrm{diag}(x_1^k, x_2^k, \ldots, x_{n_1}^k)$, $D_y = \mathrm{diag}(y_1^k, y_2^k, \ldots, y_e^k)$, and $e^T = (1, \ldots, 1)$.

(2) Compute $w' = (e/N) + (\alpha w^d / N)$, where $N = \dim(\mathrm{w})$, and α a constant.

(3) $w^{k+1} = (\sum_i c_i + 1)(Dw' / e^T Dw')$.

The solution w^d is given upto a scale factor by

$$w^d = D_x p_x + B^T (BQ^{-1}B^T)^{-1} BQ^{-1} D_x p_x$$

where
$$Q = \begin{pmatrix} I + D_x C^T D_y^{-2} C D_x & -D_x C^T D_y^{-2} c \\ -c^T D_y^{-2} C D_x & 1 + c^T D_y^{-2} c \end{pmatrix},$$

$$B = \begin{pmatrix} APD_x & 0 \\ b_1^T & d_2 \end{pmatrix}, \quad b_1 = e_x - D_x C^T D_y^{-1} e_y, \quad d_2 = 1 + e_y D_y^{-1} c$$

and $e^T = (e_x^T, e_y^T, 1)$. To express w^d in the above form, we first eliminate capacity constraints by substituting for the slacks y. This transforms the sphere $w^T w \le 1$ into the ellipsoid $(x^T, z)\, Q \begin{pmatrix} x \\ z \end{pmatrix} \le 1$, and the dimension of the constraint matrix is reduced from se to sv at the cost of slightly increasing the complexity of the quadratic constraint. The equation for w^d is then obtained using Lagrange multipliers.

We now describe how to compute the direction w^d. We note that matrices are not explicitly computed unless so stated. We shall show how to efficiently compute an expression, comprising addition, subtraction, and multiplication, of a constant number of matrices, for Q^{-1} and a similar expression for $(BQ^{-1}B^T)^{-1}$. Then the direction vector w^d may be obtained using the above equation for w^d. In the following computations we shall repeatedly use the formula [2],

$$(A + UV^T)^{-1} = A^{-1} - A^{-1}U(I + V^T A^{-1} U)^{-1} V^T A^{-1}.$$

1. Q can be expressed as $D_1 + U_1 V_1^T$ where D_1 is a block diagonal matrix, with each block of dimension at most s, and $U_1 V_1^T$ is a matrix of rank 2. Specifically, $D_1 = \begin{pmatrix} D_{11} & 0 \\ 0 & d_1 \end{pmatrix}$, where $D_{11} = I + D_x C^T D_y^{-2} C D_x$, $d_1 = 1 + c^T D_y^{-2} c$, and $U_1^T = \begin{pmatrix} 0 \ldots 0\, 1 \\ -c^T D_y^{-2} C D_x \end{pmatrix}$, $V_1 = U_1 \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

2. D_{11}^{-1} is block diagonal and can be expressed as $D_{11}^{-1} = I - D_x C^T D_y^{-1} D_w^2 D_y^{-1} C D_x$, where D_w is a diagonal matrix, in $O(s\,e)$ operations. Then D_1^{-1} is expressed as $\begin{pmatrix} D_{11}^{-1} & 0 \\ 0 & d_1^{-1} \end{pmatrix}$ in $O(N)$ extra operations.

2.a Q^{-1} can then be expressed as $Q^{-1} = D_1^{-1} - U_2 U_3 U_4^T$, where $U_2 = D_1^{-1} U_1$, $U_3^{-1} = I + V_1^T D_1^{-1} U_1$, and $U_4 = D_1^{-1} V_1$, in $O(s\,e)$ operations.

3. We next note that $BQ^{-1}B^T = BD_1^{-1}B^T + (BU_2U_3)(BU_4)^T = A_2 + U_5 V_5^T$, where $A_2 = \begin{pmatrix} A_1 & 0 \\ 0 & d_3 \end{pmatrix}$, $A_1 = APD_x D_{11}^{-1} D_x P^T A^T$, $d_3 = b_1^T D_{11}^{-1} b_1 + d_2^2 d_1^{-1}$, $U_5^T = \begin{pmatrix} 0 \ldots 0\, 1 \\ b_2^T \\ (BU_2U_3)^T \end{pmatrix}$, $V_5^T = \begin{pmatrix} b_2^T \\ 0 \ldots 0\, 1 \\ (BU_4)^T \end{pmatrix}$, $b_2^T = ((APD_x D_{11}^{-1} b_1)^T, 0)$.

4. $(BQ^{-1}B^T)^{-1}$ can now be written as $(BQ^{-1}B^T)^{-1} = A_2^{-1} - A_2^{-1} U_5 (I + V_5^T A_2^{-1} U_5)^{-1} V_5^T A_2^{-1}$. Once D_{11}^{-1} is available as described above, A_1 can be computed in $O(s^2 e)$ operations. We have $A_1 = A_{11} - A_{12}$, where $A_{11} = APD_x^2 P^T A^T$, $A_{12} = APD_x^2 C^T D_y^{-1} D_w^2 D_y^{-1} C D_x^2 P^T A^T$. APD_x is a weighted graph incidence matrix with $O(s\,e)$ entries, hence A_{11} has $O(s\,e)$ entries and is computed in $O(s\,e)$ operations. An entry in A_{12} is either the product of the weighted flows of two commodities in an edge, or for a fixed pair of commodities the sum of such products corresponding to all edges incident on a vertex. So A_{12} is computed in $O(s^2 e)$ operations. A_1^{-1} and A_2^{-1} are then computed in $O(s^3 v^3)$ more arithmetic operations.

5. $A_2^{-1}U_5$, $A_2^{-1}V_5$, and $(I+V_5^T A_2^{-1}U_5)^{-1}$ require $O(s^2v^2)$ operations once A_2^{-1} is available, and the direction vector w^d is then obtained in $O(s^2v^2+se)$ additional operations, using the computed expression for $(BQ^{-1}B^T)^{-1}$.

2.3. Reducing the amortized complexity

The complexity of Problem MF may be further reduced by using an approximation D_Δ to D in the local optimization problem solved at each iteration. Let w^k be the point at the end of the $(k-1)^{st}$ iteration. At the start of the k^{th} iteration D_Δ is computed as follows.

(1) $D_\Delta := (N^{-1}\sum_i(w_i^k/w_i^{k-1}))D_\Delta$.

(2) For $i=1,2,...,N$, if $((D_\Delta)_{ii} < w_i^k/\sqrt{2}$ or $(D_\Delta)_{ii} > \sqrt{2}\,w_i^k)$ then $(D_\Delta)_{ii} := w_i^k$.

(3) If (k is a multiple of $\lceil N^\delta \rceil$) then $D_\Delta := \mathrm{diag}(w_1^k, w_2^k, \ldots, w_N^k)$.

Modifying an element of D_Δ in Step 2 leads to a rank one change in A_1 and hence in A_1^{-1}, and this change can be computed in $O(s^2v^2)$ operations. We reset D_Δ in Step 3 since just modifying D_Δ in Step 2 leads to an excessive number of rank one changes in A_1. Whenever D_Δ is reset, we recompute A_1^{-1} as described in the previous section in $O(s^3v^3)$ operations. Once A_1^{-1} is available, the direction vector w^d can be obtained in $O(s^2v^2)$ additional operations as described in Section 2.2. We note that even though D_Δ is used instead of D, the number of iterations is still $O(NL)$ [4].

Lemma. Between successive resettings of D_Δ in Step 3, the total number of modifications to D_Δ in Step 2 is $O(N^{2\delta})$, if $\delta < 1/2$.

Proof. Let n_i be the number of times $(D_\Delta)_{ii}$ is modified in Step 2 between successive resettings in Step 3. Let $d_i^k = (N^{-1}\sum_i(w_i^k/w_i^{k-1}))\,w_i^k/w_i^{k-1}$, $h_i^k = \ln(d_i^k)$, J^k the set of those indices i such that $|d_i^k - 1| \geq (8\lceil N^\delta\rceil)^{-1}$, $\pi_i = \{k : 0 \leq k \leq \lceil N^\delta \rceil, i \in J^k\}$, $\theta_i = \{k : 0 \leq k \leq \lceil N^\delta\rceil, i \notin J^k\}$.

As $(D_\Delta)_{ii}$ is modified whenever the product of successive (d_i^k)'s exceeds $\sqrt{2}$ or falls below $1/\sqrt{2}$, we have

$$n_i \ln\sqrt{2} \leq \sum_{k=1}^{\lceil N^\delta\rceil} |h_i^k|.$$

As $\sum_{k\in\theta_i} |h_i^k| \leq 1/2$, $n_i \geq 1$ implies

$$n_i \ln\sqrt{2} \leq 2 \sum_{k\in\pi_i} |h_i^k|.$$

Also,
$$\sum_{i\in J^k} |h_i^k| \leq \sum_{i=1}^{N} |h_i^k - (d_i^k-1)| + \sum_{i\in J^k} |d_i^k - 1| \leq \frac{\beta^2}{2(1-\beta)} + 8\lceil N^\delta\rceil\beta.$$

In [4] it is shown that (i) $\sum_i (d_i^k-1)^2 \leq \beta^2$, for some constant β, (ii) $\sum_{i=1}^{N} |h_i^k - (d_i^k-1)| \leq \frac{\beta^2}{2(1-\beta)}$, and $\sum_i (d_i^k-1)^2 \leq \beta^2$ implies $\sum_{i\in J^k} |d_i^k - 1| \leq 8\lceil N^\delta\rceil\beta$.

So, $\sum_{i=1}^{N} n_i \leq \sum_{k=1}^{r} \sum_{i\in J^k} |h_i^k| = O(N^{2\delta}\beta)$.

From the Lemma it follows that the total complexity is $O(N^{1-\delta}(s\,v)^3L + N^{1+\delta}(s\,v)^2L)$ operations. Choosing $N^\delta = (s\,v)^{0.5}$ gives the desired complexity of $O(s^{3.5}v^{2.5}e\,L)$ operations as $N = O(s\,e)$.

2.4. Jumping to the optimal solution

In this section we describe how to obtain an optimal solution to the multicommodity flow problem, once we have a feasible point where the objective function value is very close to the optimal value.

Let $\epsilon < 2^{-k_1L}$, for a suitable constant k_1. Suppose we have a point $w^T = (x^T, y^T, z)$ such that

1. The objective function value at w differs from the optimum by at most ϵ.
2. Flow conservation at each vertex is violated by at most ϵ.
3. The flow of each commodity in each edge is at least ϵ.
4. The sum of the flows in each edge does not exceed its capacity by ϵ.

From w we obtain another solution w' by the following procedure. Let G_i be the subgraph induced by those edges in which the flow of commodity i is at least ϵ, and the total flow is not within ϵ of capacity. If G_i contains a cycle, we push flow of commodity i around the cycle (without increasing the objective function) till the flow of commodity i in some edge goes to zero or the total flow in some edge reaches capacity. We repeat this process till each G_i is a forest. Obtaining W' takes $O(s^2e^2)$ operations.

Define a system of linear equations as follows.

1. Include all equations defining flow conservation.
2. If the flow of commodity i in an edge is at most ϵ then equate the corresponding variable to 0.
3. If the total flow in an edge is within ϵ of capacity then equate the sum of the flow variables corresponding to the edge to its capacity.

As each G_i is a forest the matrix describing this system of linear equations has independent columns, and the system then defines an optimal vertex [2, 7] (even though it may be over determined).

2.5. Precision of arithmetic operations

We shall use k_1, k_2,.... to denote constants. The computations during each iteration are performed to a precision of k_3L bits, and at the end of each iteration, each co-ordinate of the new point obtained is rounded to the smallest multiple of 2^{-k_2L} larger than the co-ordinate. Because of rounding, at each iteration we work with a linear program where each constraint may be violated by at most ϵ. At each iteration, ϵ increases by at most $N2^{-k_2L}$, and we may choose k_2 so that $\epsilon < 2^{-k_1L}$ when we jump to the optimal solution. It is adequate to show that the condition numbers of the matrices arising in the computation are bounded by 2^{k_4L}, we can then choose an appropriate $k_3 > k_4$. It is easily seen that the frobenius norms of C^TC, AA^T and the 2-norm of c are bounded by 2^L. Moreover, each of $\|D\|_F$, $\|D^{-1}\|_F$, is at most $2^{(k_2+1)L}$. Let $\kappa(R)$ denote the condition number of R ($\kappa(R) = \|R\|_2\|R^{-1}\|_2$). By the repeated use of $\kappa(R\,R_1R^T) \leq \kappa(RR^T)\,\kappa(R_1)$, it can be shown that $\kappa(D_{11})$, $\kappa(D_1)$, $\kappa(A_1)$ and $\kappa(A_2)$ are bounded 2^{k_5L}.

As Q is obtained by intersecting the sphere $x^Tx+y^Ty+z^2 \leq 1$ with the affine space $CD_zx+D_yy-zc=0$ and then projecting onto the space of (x^T, z), $\kappa(Q) \leq \|D_y^{-2}\|_F \|c\|_2^2 \|C^TC\|_F \leq 2^{(2k_2+4)L}$.

To bound $\kappa((BQ^{-1}B^T)^{-1})$ it is then adequate to bound $\kappa(BB^T)$. We have $B = \begin{pmatrix} APD_z & 0 \\ b_1 & d_2 \end{pmatrix}$, where $b_1 = e_z - D_zC^TD_y^{-1}e_y$ and $d_2 = 1+e^TD_y^{-1}c$. Let $B_1 = \begin{pmatrix} B_{11} \\ b_1 \end{pmatrix}$, $B_{11} = APD_z$ and let $w^T = (w_1^T, w_n)$ be a unit vector. Let b_{11} be the component of b_1 in the column space of B_{11} and b_{12} the component in the orthogonal complement of the column space of B_{11}. For simplicity let us assume that during local optimization we work with a D whose entries have been rounded to powers of 2^{-k_2L} (this does not change the complexity of the algorithm). Then $\|b_{12}\|_2 \geq 2^{-k_6L}$, for some k_6, since we can find a basis for the null space of B_{11}^T with rational co-ordinates. We outline an argument to lower bound $\|B_1^Tw\|_2$. We have $B_1^Tw = B_{11}^Tw_1 + w_nb_{11} + w_nb_{12}$. As B_{11} is a graph incidence matrix with its columns weighted by entries from D_z it follows that $\|B_{11}^Tw_1\|_2 \geq 2^{-(2k_2+2)L}\|w_1\|_2$. Moreover, $\|b_{11}\|_2 \leq \|b_1\|_2 \leq 2^{(3k_2+2)L}$. We have two cases depending on the magnitude of w_n. If $w_n > 2^{-(5k_2+5)L}$ then $\|w_nb_{12}\|_2 \geq 2^{-(k_6+5k_2+5)L}$; otherwise $\|B_{11}^Tw_1+w_nb_{11}\|_2 \geq 2^{-(2k_2+2)L-1}$. So if we let $k_7 = 5k_2+k_6+6$, then both $w^T(B_1B_1^T)w$, $w^T(BB^T)w$, are at least 2^{-k_7L}, and at most $2^{(4k_2+4)L}$, and it then follows that $\kappa(BB^T) \leq 2^{(k_7+4k_2+4)L}$.

Finally, we must show how to control the error increase in A_1^{-1} during rank one updates without paying an excessive penalty in number of operations. Suppose we have an approximate inverse A_1' of A_1 such that $A_1A_1' = I+E_1$. $(A_1+uv^T)^{-1}$ is computed as follows.

1. Compute an initial approximation $A_1'' = A_1' - ((A_1'u)(A_1'v)^T/(1+u^TA_1'v))$
2. $(A_1'+uv^T)(A_1'') = I+E_1+E_2$, where E_2 is a constant rank matrix computable in $O(s^2v^2)$ operations. The required approximate inverse is $A_1''-E_2$, and $(A_1+uv^T)(A_1''-E_2) = I+E_1-E_2(E_1+E_2)$. So the error term is about E_1 if we choose k_2 and k_3 so that E_2^2 is of lower order than E_1.

3. Conclusions

We have extended Karmarkar's interior point method to give an efficient algorithm for Convex Quadratic Programming. This approach is also applicable to minimizing other convex functions over polytopes provided these functions can be efficiently minimized over ellipsoids. We have also described how Karmarkar's algorithm can be speeded up for the Multicommodity Flow problem. The technique used here extends in a straightforward manner to similarly structured linear programs.

Acknowledgements

The authors would like to thank Prof. C. L. Liu, Prof. E. M. Reingold, and M. Wong for fruitful discussions. The authors would also like to thank S. Ashby, A. Chronopolous, J. Purtilo, and Prof. P. Saylor for discussions on related topics.

Appendix

Sliding Objective Function Method

We shall show that the number of iterations in the algorithm in Section 1.3 is bounded by $O(NL)$. Let LOW_j, $HIGH_j$, and u_j, denote the value LOW, $HIGH$, and the threshold u, during the jth stage. At the jth stage we measure potential w.r.t the guess g_j where $g_j = LOW_j + c_1(HIGH_j - LOW_j)$. Moreover, $u_j = LOW_j + c_2(HIGH_j - LOW_j)$. Let z^j be the point at the beginning of the jth stage.

Case 1. $g_{j+1} > g_j$.

Then $LOW_{j+1} = g_j$, $HIGH_{j+1} = HIGH_j$.

Case 1.1. $f(z^{j+1}) \leq f(z^0)$.

$$\frac{f(z^{j+1})-g_{j+1}}{f(z^0)-g_{j+1}} \leq \frac{f(z^{j+1})-g_j}{f(z^0)-g_j}$$

Case 1.2. $f(z^{j+1}) > f(z^0)$.

Then $g_{j+1} = g_j + c_1(HIGH_{j+1} - g_j)$, and $f(z^0)-g_{j+1} \geq (f(z^0)-g_j)(1-c_1)$ since $f(z^0) \geq HIGH_{j+1}$. Hence

$$\frac{f(z^{j+1})-g_{j+1}}{f(z^0)-g_{j+1}} \leq \frac{f(z^{j+1})-g_j}{f(z^0)-g_j}\frac{1}{(1-c_1)}$$

Case 2. $g_{j+1} \leq g_j$.

This case occurs when the objective function value becomes less than or equal to the threshold u_j. We may assume that $HIGH_{j+1} = u_j$; otherwise we proceed through a sequence of dummy stages $j+1, \ldots, j+q$, such that $HIGH_{j+1} = u_j$, ..., $HIGH_{j+q-1} = u_{j+q-2}$, $HIGH_{j+q} \geq u_{j+q-1}$. For each of these stages we have $HIGH_{j+i+1} = f(z^{j+i+1}) \geq u_{j+i}$, $LOW_{j+i+1} = LOW_{j+i}$, and $g_{j+i+1} < g_{j+i}$. Hence,

$$\frac{f(z^{j+i+1})-g_{j+i+1}}{f(z^0)-g_{j+i+1}} \leq \frac{f(z^{j+i+1})-g_{j+i+1}}{f(z^0)-g_{j+i}}$$

$$\leq \frac{f(z^{j+i+1})-g_{j+i}}{f(z^0)-g_{j+i}}\frac{c_2(1-c_1)}{(c_2-c_1)}$$

Suppose at each iteration in the jth stage we get a decrease of at least δ in the potential $\sum_{i=1}^{N}\ln(\frac{f(z)}{z_i})$. Then after j stages we have

$$n\ln(\frac{f(z^{j+1})-g_{j+1}}{f(z^0)-g_{j+1}}) - \sum_i \ln(\frac{z_i^{j+1}}{z_i^0})$$

$$\leq -\sum_{l=1}^{j} s_l\delta + n\,j\max(-\ln(1-c_1), \ln(\frac{c_2(1-c_1)}{(c_2-c_1)}))$$

where s_l is the number of iterations in the lth stage. Within $O(L)$ stages the difference $HIGH-LOW$ falls to $2^{-\theta(L)}$ and by the above inequality the number of iterations in $O(L)$ stages is $O(NL)$. Once $HIGH-LOW$ is $2^{-\theta(L)}$, we have a good enough guess for the optimum value f_0 of $f(z)$, and we generate a sequence of points keeping $HIGH$, LOW, and the guess for f_0, fixed, till we obtain a point where the objective function value is at most $2^{-\theta(L)}$ away from $HIGH$.

References

1. Chvatal, V., *Linear Programming,* W. H. Freeman, New York, 1983.
2. Golub G. H., Van Loan C. F., *Matrix Computations,* The Johns Hopkins University Press, Baltimore, MA, 1983.
3. Hu, T. C., *Integer Programming and Network Flows,* Addison-Wesley, Reading, M.A., 1969.
4. Karmarkar, N., "A New Polynomial-Time Algorithm for Linear Programming," *Combinatorica,* **4** (4), (1984), 373–395.
5. Khachian, L.G., "Polynomial algorithms in linear programming," *USSR Computational Mathematics and Mathematical Physics,* **20** ,(1979), 191–194.
6. Kozlov, M. K., Tarasov, S. P., and Khachian, L. G., "Polynomial solvability of convex quadratic programming," *Dokl. Akad. Nauk SSSR,* **5** ,(1979), 1051–1053.
7. Papadimitriou C. H. and Steiglitz K., *Combinatorial Optimization: Algorithms and Complexity,* Prentice-Hall, Englewood Cliffs, New Jersey, 1982.
8. Stewart G. W., *Introduction to Matrix Computations,* Academic Press, Inc., New York, 1973.
9. Wilkinson J. H., *The Algebraic Eigenvalue Problem,* Oxford University Press, 1965.
10. Zoutendijk, G., *Mathematical Programming Methods,* North-Holland, New York, 1976.