A Report
on
Interior Point Methods

By

Thejeswar Reddy Narravala

A20404518

Interior Point Methods:

Introduction:

The Simplex method solves the linear programming problems by moving from one basic feasible solution to another BFS while improving the cost of objective function.

In the mid of 1980, new methods are introduced to solve the linear programming problems by moving inside the feasible region.

Since the these algorithms are trying to find an optimal solution while moving inside the feasible set, it is called Interior point methods.

The Interior point methods are shown to have better performance while solving the large linear programming problems compared to simplex methods.This is because the number of BFS can be very large and takes large amount of time to solve using simplex.

The Interior point methods combine the advantages of the simplex method and of the ellipsoid algorithm and work in polynomial time.

The Affine Scaling Algorithm:

In this algorithm, we approximate a polyhedron by an ellipsoid.The approximating ellipsoid is contained in the polyhedron.

Consider the following Linear Program:

Primal	Dual
$Min\ C^T x$	$Max\ P^T b$
$S.T\ \ Ax = b$	$S.T\ \ P^T A \leq\ C^T$
$x \geq 0$	$P\ is\ free$

Geometric Idea:

Let $P = \{ x \mid Ax = b, x \geq 0 \}$ be a feasible set and $\{ x \in P \mid x \geq 0 \}$ is the interior of P and its elements are called Interior points.

Here,Instead of minimizing the $C^T x$ over entire polyhedron, we try to optimize the cost function $C^T x$ over an ellipsoid and solution can be obtained in closed form.

So, we solve a series of optimization problems over ellipsoids.

We start with feasible solution $x^o > 0$, where $\{x \in P \mid x > 0\}$ we form an ellipsoid S^o with centre as x^o.

Now, we optimize the cost function over all $x \in S^o$ and find a point x^1 and proceed to optimize over all $x \in S^1$ and find a point x^2 and proceed to optimize over all $x \in S^2$ and the process continues until we cannot find better solution.

Geometric representation:

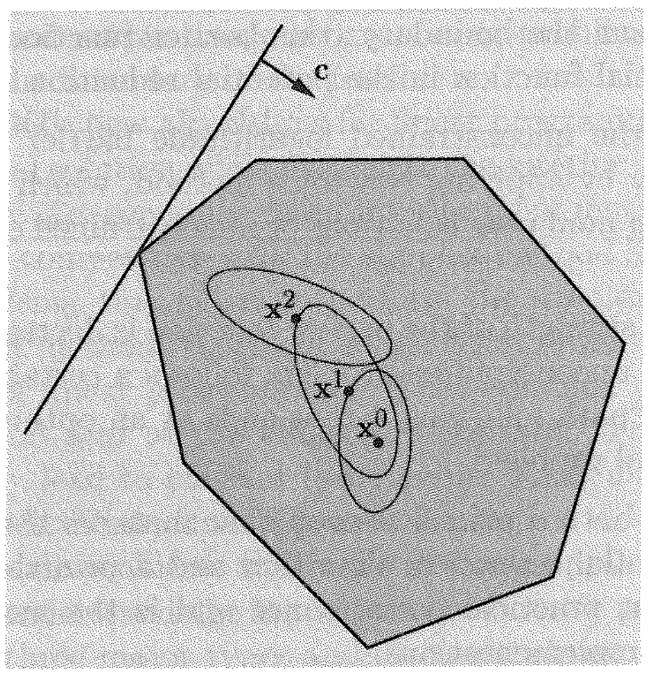

Figure 9.1: Illustration of the affine scaling algorithm. The vector $\mathbf{x}^1$ minimizes $\mathbf{c}'\mathbf{x}$ over all $\mathbf{x}$ in the ellipsoid centered at $\mathbf{x}^0$. The vector $\mathbf{x}^2$ minimizes $\mathbf{c}'\mathbf{x}$ over all $\mathbf{x}$ in the ellipsoid centered at $\mathbf{x}^1$.

Lemma: 9.1

Let $\beta \in (0,1)$ be a scalar, let $y \in \mathbb{R}^n$ satisfy $y > 0$ and let

$$S = \{\ x \in \mathbb{R}^n \mid \sum_{i=1}^{n} (x_i - y_i)^2 \div y_i^2 \leq \beta^2\}$$ *Then , $x > 0$ for every $x \in S$.*

Proof:

Let $x \in S$,
We then have,

$(x_i - y_i)^2 \leq \beta^2 y_i^2 < y_i^2$ $\forall\, i$ *from lemma statement*
We can write, left side equation as,

$$|x_i - y_i| \leq \beta y_i \rightarrow \quad (1)$$

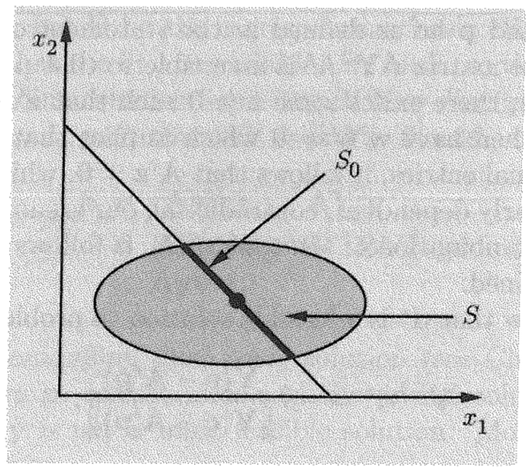

Figure 9.2: The sets S and S_0 for the case where $n = 2$ and there is a single equality constraint $x_1 + x_2 = 1$.

$$|x_i - y_i| < y_i \rightarrow \quad (2)$$

We can write (2) as , $-x_i + y_i < y_i$

$$\implies x_i > 0$$

Now, for some $y \in \mathbb{R}^n$, $S.T\ y > 0$

$$Ay = b$$

Let's assume $Y = \begin{bmatrix} y_1 & 0 & 0 \\ 0 & . & 0 \\ 0 & & y_n \end{bmatrix} = diag(y_1, \ldots., y_n)$

Since all its diagonal elements are positive we can say Y *is invertible.*
Now, from (1), we can write as,

$$|x_i - y_i| \leq \beta y_i$$

$$\implies \| Y^{-1}(x - y) \| \leq \beta$$

||.|| represents Euclidean norm.

The set S is an ellipsoid centered at y *.*
We can say S_0 *itself is an ellipsoid contained in the feasible set.*

$$S_0 = S \cap \{x | Ax = b\}$$

Now, the next step is to replace the original linear programming problem with minimization problem over S_0 ellipsoid.

i.e… $\quad Min\ C^T x$

$S.T\ Ax = b$

$\| Y^{-1}(x - y) \| \leq \beta$ *(* this represents $x \in S$ *)*

Let's introduce new variables d = $x - y$

We already know, $y \in \mathbb{R}^n$*,* $y > 0\ \ S.T\ \ Ay = b$

We also know, $x \in S_0$

$\implies Ax = b$

$d = x - y$

$\implies Ad = Ax - Ay$

$\implies Ad = b - b = 0 \implies Ad = 0$

Let's write a new LP to optimize over d,

i.e…

LP (I):

$$Min\ C^T d$$

$$S.T$$

$$Ad = 0$$

$$|Y^{-1}d| \leq \beta$$

Lemma 9.2

Assume that the rows of A are linearly independent and that C is not a linear combination of the rows of A. Let y be a positive vector.Then, an optimal solution d to the problem (I) is given by*

$$d^* = -\beta \frac{Y^2(C - A^T P)}{||Y(C - A^T P)||} \text{ Where } \quad P = (AY^2A^T)^{-1}AY^2C$$

Furthermore, the vector $x = y + d^*$ *belongs to P and*

$$C^T x = C^T y - \beta ||C - A^T P|| < C^T y$$

Lemma 9.3

Let y and p be a primal and a dual feasible solution respectively, such that

$$C^T y - b^T p < \epsilon$$

Let y^* *and* p^* *be optimal and dual solutions respectively.*

Then,

$$C^T y^* \leq C^T y < C^T y^* + \epsilon,$$

$$b^T p^* - \epsilon < b^T p \leq b^T p^*$$

Proof:

we know y is feasible solution and y^* is an optimal solution.

Therefor, we can say $C^T y^* \leq C^T y \quad \rightarrow \quad$ *(1)*

We p is a feasible solution to the duals by week duality, we can say,

$$p^T b \leq C^T y^*$$

& we know from the given in the lemma, i.e.… $c^T y - b^T p < \epsilon$

$$\implies c^T y < b^T p + \epsilon$$

So now, we have $c^T y < b^T p + \epsilon \leq c^T y^* + \epsilon \quad \rightarrow \quad$ *(2) (since* $p^T b \leq c^T y^*$*)*

Therefore, we have showed from (1) & (2) ,

$$c^T y^* \leq c^T y < c^T y^* + \epsilon$$

Now, we know from strong duality, if p^* *and* y^* *are optimal solutions to dual and primal respectively, then we can say* $b^T p^* = c^T y^*$ *and*

we know, $c^T y^* \leq c^T y$
$\implies b^T p^* \leq c^T y$

and we also know $c^T y < b^T p + \epsilon$ *from (2)*

We can say , $b^T p^* \leq c^T y < b^T p + \epsilon$

$b^T p \leq b^T p^*$ *(since it is maximization and* p^* *is optimal solution) ————————-(3)*

Since to prove $b^T p^* - \epsilon < b^T p$*, we know* $c^T y < b^T p + \epsilon$ *from given*

We also know, $c^T y^* \leq c^T y$

$$\implies c^T y^* < b^T p + \epsilon$$
$$\implies b^T p^* < b^T p + \epsilon$$
$$\implies b^T p^* - \epsilon < b^T p$$
$$\implies b^T p^* - \epsilon < b^T p^* \text{ ——————————-(4)}$$

From (3) & (4), We can say,

$$b^T p^* - \epsilon < b^T p \leq b^T p^*$$

Interpretation of the formula for P:

In lemma 9.2, we assumed some $y > 0$, now let us examine what happens if we let y be a non-degenerate basic feasible solution and apply the same formula to define a vector P.

Let B be a corresponding basis.

Let's assume that the first m variables are basic, i.e.... A = [B N] N has dimensions m x (n-m).

(Since (n-m) non basic variables with m constraints).

If $y = diag(y_1, \ldots, y_m, 0, \ldots, 0)$ and $y_o = diag(y_1, \ldots, y_m)$

Then AY = [By_o 0] & P = $(AY^2A^T)^{-1}AY^2c$

$$
\begin{aligned}
&= (B^TY_o^2B)^{-1}bY_oc_B \\
&= (B^T)^{-1}Y^{-2}B^{-1}BY_o^2c_B \\
&= (B^T)^{-1}c_B
\end{aligned}
$$

So P = $(B^T)c_B$ is the corresponding dual solution.

Now, the vector r = $c - A^Tp$ (Since optimality) becomes $r = c - A^Tc_B(B^T)^{-1}$ (since reduced cost in the simplex method).

Note:

If Y is degenerate , then the matrix AY^2A^T is not invertible and this interpretation breaks down. So, we will assume that all primal basic feasible solutions are non-degenerate.

Suppose, r = $c - A^Tp$ is non-negative, then it is dual feasible solution (i.e... p is dual feasible).

$$
\begin{aligned}
&\implies r = c - A^Tp \\
&\implies r^Ty = (C - A^Tp)y \\
&\implies r^Ty = c^Ty - A^Typ \\
&\implies r^Ty = c^Ty - b^Tp
\end{aligned}
$$

i.e... the difference in objective values between the primal solution y and the dual solution p is simply r^Ty.

Therefore,r^Ty is called Duality Gap.

By weak duality, the duality gap is always non-negative.
If $r^T y = 0$, then the complementary slackness conditions or strong duality hold and the vectors y and p are primal and dual optional solutions respectively.

If the duality gap satisfies $r^T y < \epsilon$, where $\epsilon > 0$ is small, the. We can say the primal and the dual solutions are "near - optimal."

We know that algorithm terminates when $r \geq 0$ (i.e… dual feasibility and primal optimality) and the dual gap $r^T y = y^T r = e^T yr$ is small,

E = [1 1 . . . 1] then,

the primal and dual solutions y and p are "near - optimal". Which means their cost is within ϵ from optimal cost.

Affine Scaling Algorithm:

Let's assume k is an integer variable used to index successive iterations.
ϵ is non-negative constant, which us used measure the closeness to optimality.
The vector X^k represent y, and the matrix X_k represents y.

The affine scaling algorithm uses the following inputs:

(a) The data if the problem(A, b, c)

(b) An initial primal feasible solution $X^0 > 0$,which can be found using auxiliary primal LP.

(c) The optimality tolerance $\epsilon > 0$

(d) The parameter $\beta \in (0,1)$

Initialization:

For Affine scaling algorithm to start, we need to have X^0,i.e… we need an interior feasible solution. We can construct such a feasible solution by using a Big M-method.

Let e $\in \mathbb{R}^n$ be the vector with all components equal to 1. E = (1 1 . . . 1). We introduce slack variables x_{n+1}.
We create a new column $A_{n+1} = b - Ae$
Let's consider the below LP,

$$Min\ c^T x + Mx_{n+1}$$
$$\text{S.T}\ \ Ax + (b - Ae)x_{n+1} = b$$
$$(x, x_{n+1}) \ \ \geq 0$$

Where M is a large positive scalar.$(x, x_{n+1}) = (e,1)$ is also a positive feasible solution to the augmented problem and the affine scaling algorithm can be applied.

If M is very large and as long as the original problem has an optimal solution, it can be shown that optimal solution to the augmented problem will have $x_{n+1} = 0$ and therefore we can have optimal solution to the original problem.

Iterative Procedure for the primal affine scaling algorithm

Step1: (Initialization) Set k=0 and find $x^0 > 0$ such that $Ax^0 = b$

Step2: (Computation of dual estimates) Compute the vector of dual estimates $p^k = (Ax_k^2A^T)^{-1}Ax_k^2c$

Step3: (Computation of reduced costs) Calculate the reduced costs vector $r^k = c - A^Tp^k$

Step4: (Check for Optimality) If $r^k \geq 0$ and $e^TX_kr^k \leq \epsilon$(a given small positive number) then stop. x^k is primal optimal and p^k is dual optimal. Otherwise, go to next step.

Step5: (Obtain the direction of translation) Compute the direction $d_y^k = -X_kr^k$

Step6: (Check for unboundedness and constant objective value) If the $d_y^k > 0$, then stop.The problem is unbounded. If the $d_y^k = 0$, then also stop.x^k is primal optimal.Otherwise go to step7.

Step7: (Compute step-length) compute the step length $\beta_k = Min \{ \frac{\beta}{-(d_y^k)_i} \mid (d_y^k)_i < 0 \}$

$\beta \in (0,1)$

Step8: (Move to a new solution) perform the translation

$$x^{k+1} = x^k + \beta_kX_kd_y^k$$

Reset the k ← k + 1 and go tp step2 and do the same steps until the algorithm terminates.

Example: *Minimize* $-2x_1 + x_2$

Subject to : $x_1 - x_2 + x_3 = 15$

$$x_2 + x_4 = 15$$
$$x_1, x_2, x_3, x_4 \geq 0$$

So, we have $A = \begin{bmatrix} 1 & -1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$ $\quad b = \begin{bmatrix} 15 \\ 15 \end{bmatrix}$ $\quad c = \begin{bmatrix} -2 \\ 1 \\ 0 \\ 0 \end{bmatrix}$

Let us start with, $x^o = [10 \quad 2 \quad 7 \quad 13]^T$, which is an interior feasible solution.

$$X_o = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 13 \end{bmatrix}$$

& $$p^o = (AX_o^2A^T)^{-1}AX_0^2x$$

After calculating,

$$p^o = [-1.33353 \quad -0.00771]^T$$

Now we need to calculate the reduced costs. i.e…

$$r^o = c - p^oA^T = \begin{bmatrix} -2 \\ 1 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 \\ -1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -1.33353 \\ -0.00771 \end{bmatrix}$$

$$= \begin{bmatrix} -0.66647 \\ -0.32582 \\ 1.33535 \\ -0.00771 \end{bmatrix}$$

From above, we can say, some components of r^0 are negative.Now we need to calculate $e^TX_or^o$.

i.e… $e^TX_or^o = [1 \quad 1 \quad 1 \quad 1] \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 13 \end{bmatrix} \begin{bmatrix} -0.66647 \\ -0.32582 \\ 1.33535 \\ -0.00771 \end{bmatrix}$

$= 2.1187$

Since some components of r^o are negative and $e^T X_0 r^0 = 2.1187$, we know that current solution is not optimal.
Therefore, we need to calculate the direction of translation with,

$$d_y^k = -X_k r^k = \begin{bmatrix} 6.6647 \\ 0.6516 \\ -9.3475 \\ 0.1002 \end{bmatrix}$$

Suppose, that $\beta = 0.99$ is chosen then step-length

$$\beta_0 = \frac{0.99}{9.3475} = 0.1059$$

Therefore, the new solution is

$$X^{k+1} = x^k + \beta_k X_k d_y^k$$

$$X^1 = X^o + \beta_o X_o d_y^o$$

$$= \begin{bmatrix} 10 \\ 2 \\ 7 \\ 13 \end{bmatrix} + 0.1059 \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 13 \end{bmatrix} \begin{bmatrix} 6.6647 \\ 0.6516 \\ -9.3475 \\ 0.1002 \end{bmatrix}$$

We can see, the objective function value has been improved from -18 to -31.99822. we can continue the iterations further and eventually leads to optimal solution.

$x^* = [30 \quad 15 \quad 0 \quad 0]^T$ with optimal value -45.

Computational Performance:

At each iteration, the computational bottleneck is the calculation of the dual estimates
$p^k = (AX_k^2 A^T)^{-1} AX_k^2 c$---------->$O(m^2 n)$
we then solve a system of linear equations involving the matrix
$AX_k^2 A^T$----------------->$O(m^3)$

In total, each step of the algorithm needs $O(m^2 n + m^3)$ arithmetic operations per iteration is $O(n^3)$.

- If algorithm start with x^o which is very nearer to BFS, then it is observed that the algorithm tends to travels along the boundaries of the feasible set.

- So,It is believed that running time of algorithm is exponential in the worst case scenario.

- If the current point is near the boundary of the feasible set, the approximating ellipsoids can be very small and the algorithm takes small steps.

- If the current point lies "deep" inside the feasible set, the approximating the ellipsoids are large and the algorithm makes rapid progress.Indeed, it has been observed that the objective function values decrease very fast in early iterations, but the rate of decrease slows down considerably while approaching the boundary.

The Potential reduction algorithm:

This interior point algorithm for linear programming only requires a polynomial number of iterations.

The algorithm solves the linear programming problem,

minimize $c^T x$ maximize $p^T b$
subject to. $Ax = b$ $\rightarrow$ subject to. $p^T A + S^T = c^T$
$x \geq 0$ $p \; is \; free, \quad S \geq 0$

With the following assumption.

Assumption: The matrix A has linearly independent rows and there exist $x > 0$ and (p, S) with S > 0, which are feasible for the primal and the dual problem, respectively.

The issue that need to be addressed in the affine scaling algorithm is the algorithm slows down while approaching the boundary of feasible set.

One possible way to address this issue is to "repel" the current solution away from the boundary of the feasible set, and then the algorithm can make the significant progress in the future steps.

Intuitively, we want an algorithm that moves inside the feasible set while staying away from the boundary of feasible set and also decreasing objective function value at each step.

In order to address above objectives, we introduce the potential function $G(x, S)$ defined by,

$$G(x, S) = q \; log S^T x - \sum_{j=1}^{n} log x_j - \sum_{j=1}^{n} log s_j$$

q is a constant larger than n(the number of variables).If x and (p,S) are primal and dual feasible solutions, respectively.

The first term in the $G(x, S)$ is a measure of the duality gap(It is the difference between the primal and dual solutions).Because,

$$\begin{aligned} c^T x - b^T p &= (S^T + p^T A)x - (A^T x^T)p \\ &= S^T x + p^T A x - pA^T x^T \\ &= S^T x \end{aligned}$$

The second and third terms penalize proximity to the boundary of the feasible sets for the primal and dual respectively (Since we have boundaries on x and S i.e… $x \geq 0$,$S \geq 0$).

Theorem 9.4:

Let $x^o > 0$ and (p^o, S^o) with S^o, be feasible solution to the primal and dual problem, respectively. Let $\epsilon > 0$ be the optimality tolerance.Any algorithm that maintains primal and dual feasibility and reduces $G(x, S)$ by an amount greater than (or) equal to $\delta > 0$ at each iteration, finds a solution to the primal and dual problems with duality gap.

$$(S^K)^T x^k \leq \epsilon,$$

after, $k = \lceil \dfrac{G(x^o, S^o) + (q-n)log(\frac{1}{\epsilon}) - n \ longn}{s} \rceil$ iterations.

Proof:

The potential function is $G(x, S) = q \ logS^T x - \sum_{j=1}^{n} logx_j - \sum_{j=1}^{n} logs_j$

$$= n \ logS^T x + (q-n) \ logS^T x - \sum_{j=1}^{n} logx_j - \sum_{j=1}^{n} logs_j$$

We can say, $n \ logS^T x - \sum_{j=1}^{n} logx_j - \sum_{j=1}^{n} logs_j$ has minimum value at $x_j s_j = \dfrac{S^T x}{n}$. We can verify by setting derivative to zero and also checking the derivative is non-negative.

Therefore, $n\ logS^Tx - \sum_{j=1}^{n} logx_j - \sum_{j=1}^{n} logs_j \geq n\ logn$

$$\Longrightarrow G(x,S) \geq n\ logn + (q-n)logS^Tx ----------(1)$$

For some, $\delta > 0$, let's assume,

$$G(x^{k+1}, S^{k+1}) - G(x^k, S^k) \leq -\delta,\ \ \forall k$$

After k steps we have,

$$G(x^k, S^k) - G(x^o, S^o) \leq -k\delta$$

we get , $G(x^k, S^k) \leq -(q-n)log\frac{1}{\epsilon} + n\ logn -----------(2)$

Since $k = \lceil \frac{G(x^o, S^o) + (q-n)log(\frac{1}{\epsilon}) - n\ longn}{s} \rceil$ stated in the theorem.

Therefore, from (1) and (2) ,we get

$$(S^K)^Tx^k \leq \epsilon$$

i.e… we can bring the duality gap below the desired tolerance ϵ with k iterations.

The above theorem help us to devise an algorithm that decreases the potential function $G(x,S)$ by a constant amount at each step. That is why this is called potential reduction algorithm.

Intuitively, the idea is to start with a primal feasible solution $x > 0$ and a dual feasible solution with $S > 0$,we try to find a direction d, such that

$G(x+d,S) < G(x,S)$ and also satisfying Ad = 0 and $||X^{-1}d|| \leq \beta$ and $\beta \in (0,1)$

Which says the new point x+d is feasible.

Now , to minimize $G(x+d,S)$ with the constraints above is difficult non-linear optimization problem.So, we take the first order Taylor series of non-linear potential function $G(x+d,S)$ in d's expansion.

i.e… Min $\nabla G(x,S)^Td$

subject to. $Ad = 0$

$$||X^{-1}|| \leq \beta$$

$$\beta < 1$$

The above LP is similar to the one discussed in affine scaling except for objective function.

Instead of c we have $\hat{c}$ = $\nabla_x G(x, S)$

i.e… $c_i = \dfrac{dG(x,S)}{dx_i} = \dfrac{qS_i}{S^T x} - \dfrac{1}{x_i}$ ----------(1)

By applying lemma 9.2, and setting Y = X and c = $\hat{c}$, we get optimal direction as

$$d^* = -\beta X \frac{u}{||u||}$$ --------> eq(a)

$u = X(\hat{c} - A^T(AX^2A^T)^{-1}AX\hat{c})$

By multiplying x with $\hat{c}$,

We get $X\hat{c} = \dfrac{q}{S^T x} XS - e$

We obtain, u = ($I - XA^T(AX^2A^T)^{-1}AX)(\dfrac{q}{S^T x}XS - e$), I is identity matrix

Moreover, $G(x, S)$ decreases by $\beta||u|| + O(\beta^2)$, where the first term comes from eq(a), second term is for the omitted higher order terms in the Taylors series expansion of $G(x, S)$.

If $||u||$ is greater than a certain threshold γ, then the optimal function decreases by at least a constant amount.

The Potential reduction algorithm uses the following inputs:

(a) The data of the problem (A, b, c);the matrix A is assumed to have full row rank.

(b) The initial primal and dual feasible solutions $x^o > 0, S^o > 0, P^o$

(c) The optimality tolerance $\epsilon > 0$

(d) The parameters β, γ, q

Potential Reduction Algorithm:

Step1:Start with some feasible solution $x^o > 0, S^o > 0, P^o$ and set k=0

Step2:(Optimality test) if $(S^k)^T X^k < \epsilon$ stop; else go to step3.

Step3:(Computation of update direction)

let $X_k = diag(x_1^k, \ldots., x_n^k)$,

$$\hat{A} = (AX_k)^T(AX_k^2A^T)AX_k,$$

$$u^k = (I - \hat{A})(\frac{q}{(S^k)^T X^k)} X_k S^k - e),$$

$$d^k = -\beta X_k \frac{u^k}{||u^k||}$$

Step4: Primal step if $\|u^k\| \geq \gamma$, then let

$$X^{k+1} = x^k + d^k,$$

$$S^{k+1} = s^k$$

$$P^{k+1} = P^k$$

Step5: Dual step if $\|u^k\| < \gamma$, then let

$$X^{k+1} = x^k,$$

$$S^{k+1} = \frac{(s^k)^T X^T}{q}(X_k)^{-1}(u^k + e)$$

$$P^{k+1} = P^k + (AX_k^2A^T)^{-1}AX_k\ (X_k S^k - \frac{(S^k)^T X^k e}{q})$$

Step6: Let k := k+1 and go to step2.

Primal Path following Algorithm

In this algorithm we deal with non negative inequalities constraints with barrier functions that prevents any variable from reaching the boundary ($x_j = 0$).

We will ensure that the variable does not reach boundary by adding $-log\ x_j$ to the objective function.

So, when x_j tends to zero it goes undefined.

So, we introduce the following barrier function, let $\mu > 0$

$$B_\mu(x) = c^T x - \mu \sum_{j=1}^{n} log\ x_j$$

If $x_j \leq 0$ for some j, the barrier function is defined to be infinity.

Let’s consider the following family of non-linear program problems called barrier problems, parameterized by $\mu > 0$.

LP(1)

Min $B_\mu(x)$

S.T $Ax = b$

$\forall\ \mu > 0$, the barrier problem has an optimal solution denoted by $X(\mu)$.It is easy to show that the above LP cannot have multiple solutions because the barrier function is “Strictly Convex”.

As μ varies, the minimizers $x(\mu)$ form the central path.

It can be show that $lim_{\mu \to 0} x(\mu)$ exists and is an optimal solution x^* to the initial linear programming problem.Because when μ is very small, the logarithmic term is negligible almost every where, except that it still prevents us from reaching the boundary.

The dual of the barrier problem is,

LP(2)

Maximum $P^T b + \mu \sum_{j=1}^{n} log\ S_j$

S.T $\quad P^T A + S^T = c^T$

Let $P(\mu)$ and $S(\mu)$ be optimal solutions to the above problem for $\mu > 0$.

The optimal solution to the below Linear program is called analytical center of the feasible set.

Min $-\sum_{j=1}^{n} log\ S_j$, S.T $Ax = b$

Lemma 9.5

If x^*, p^* and s^* satisfy Karush-kuhn-Tucker optimality conditions then they are optimal solutions to the problem LP(1) and LP(2) i.e..

$x^* = x(\mu), p^* = p(\mu)\ , s^* = s(\mu)$

Karush-kuhn-Tucker optimality conditions

$Ax(\mu) = b$

$x(\mu) \geq 0$

$A^T p(\mu) + s(\mu) = c$

$s(\mu) \geq 0$

$x(\mu)s(\mu)e = e\mu$

Example1: A simple barrier problem

Consider the following linear programming problem,

Minimize x

S.T $x \geq 0$

The barrier function in this case is $B_\mu(x) = x - \mu log\ x$

The find the minimizer take derivative and set to zero.

i.e… $1 - u\dfrac{1}{x} = 0 \implies x(\mu) = \mu$

We can say that as μ decreases to 0, the optimal solution approaches $x^* = 0$.

Example2:

Min x_2

S.T $x_1 + x_2 + x_3 = 1$

$x_1, x_2, x_3 \geq 0$

Let P be feasible set,

The barrier function is Min $x_2 - \mu log\ x_1 - \mu log\ x_2 - \mu log\ x_3$

S.T $x_1, x_2, x_3 \geq 0$

Now substitute the $x_3 = 1 - x_1 - x_2$ in the objective function

$$\implies Min \quad x_2 - \mu log\ x_1 - \mu log\ x_2 - \mu log\ (1 - x_1 - x_2)$$

By taking derivatives and setting equal to 0, we get,

$$x_1(\mu) = \frac{1 - x_1(\mu)}{2}$$

$$x_2(\mu) = \frac{1 + 3\mu - \sqrt{1 + 9\mu^2 + 2\mu}}{2}$$

$$x_3 = \frac{1 - x_2(\mu)}{2}$$

These solutions dorms the central path as μ varies.

If we let $\mu \to \infty$ the analytical center can be found at $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.

If we let $\mu \to 0$ the analytical center is the point $(\frac{1}{2}, 0, \frac{1}{2})$ for the polyhedron

$$Q = \{ x \,|\, x = (x_1, 0, x_3), x_1 + x_3 = 1, x \geq 0 \}$$