# Linear Programming

Chongnan Li
16271221@bjtu.edu.cn

**Lecture 6 Interior Point Method**

***Outline:***
***1.Motivation***
***2.Basic concepts***
***3.Primal affine scaling algorithm***
***4.Dual affine scaling algorithm***

- **How to find an initial feasible solution?**
- Big-M method

$$(\text{LP})\begin{cases}\min & \boldsymbol{c}^T\boldsymbol{x}\\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}\\ & \boldsymbol{x}\geq\boldsymbol{0}\end{cases}$$

Idea: add an artificial variable with a big penalty.

$$(\text{big-M})\begin{cases}\min & \boldsymbol{c}^T\boldsymbol{x}+Mx^a\\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x}+(\boldsymbol{b}-\boldsymbol{A}\boldsymbol{e})x^a=\boldsymbol{b}\\ & \boldsymbol{x}\geq\boldsymbol{0},\ x^a\geq 0\end{cases}$$

**Properties of (big-M) problem:**

(1) It is a standard form LP with $n+1$ variables and $m$ constraints.

(2) $\boldsymbol{e}=(\underbrace{1,1,\cdots,1}_{n+1})^T\in R^{n+1}$ is an interior feasible solution of (big-M).

(3) If $(x^a)^*>0$ in $(\boldsymbol{x}^*,(x^a)^*)$ then (LP) is infeasible. Otherwise, either (LP) is **unbounded below** or $\boldsymbol{x}^*$ is optimal to (LP).

(4) Not like big-M method in simplex, **$\boldsymbol{b}\geq\boldsymbol{0}$ is not required!**

- Two-phase method

$$(\text{LP})\begin{cases}\min & \boldsymbol{c}^T\boldsymbol{x}\\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x}=\boldsymbol{b}\\ & \boldsymbol{x}\geq\boldsymbol{0}\end{cases}$$

Choose any $\boldsymbol{x}^0>\boldsymbol{0}$, calculate $\boldsymbol{v}=\boldsymbol{b}-\boldsymbol{A}\boldsymbol{x}^0$.
If $\boldsymbol{v}=\boldsymbol{0}$, then $\boldsymbol{x}^0$ is interior feasible.

Otherwise, consider

$$\text{(Phase - I)} \begin{cases} \min & u \\ \text{s.t.} & \boldsymbol{A}\boldsymbol{x} + \boldsymbol{v}u = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0},\ u \geq 0 \end{cases}$$

**Properties of (Phase - I) problem:**

(1) (Phase - I) is a standard form LP with $n + 1$ variables and $m$ constraints.

(2) $\hat{\boldsymbol{x}}^{\boldsymbol{0}} = \begin{pmatrix} \boldsymbol{x}^0 \\ u^0 \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}^0 \\ 1 \end{pmatrix}$ is interior feasible for (Phase - I).

(3) (Phase - I) is bounded below by 0.

(4) Apply primal-affine scaling to (Phase - I) will generate $\begin{pmatrix} \boldsymbol{x}^* \\ u^* \end{pmatrix}$ for (Phase - I).

(5) If $u^* > 0$, (LP) is infeasible. Otherwise, $\boldsymbol{x}^*$ is an initial feasible solution for (LP).

- Facts of the primal affine scaling algorithm

(1) The convergency proof, i.e.,

$$\{\boldsymbol{x}^k\} \to \boldsymbol{x}^*$$

under Non-degeneracy assumption (Theorem 7.2) is given by Vanderbei/Meketon/Freedman in (1985).

(2) Convergence proof without Non-degeneracy assuption, T.Tsuchiya(1991), P.Tseng/Z.Luo(1992).

(3) The computational bottleneck is to find $(\boldsymbol{A}X_k^2\boldsymbol{A}^T)^{-1}$.

(4) No polynomial-time proof.

-J.Lagarias showed primal affine is only of super-linear rate.

-N.Megiddo/M.Shub showed that primal affine scaling **might visit all vertices** if it moves **too close to the boundary**.

**"Although in practice the primal affine scaling algorithm performs very well, no proof shows the algorithm is a polynomial-time algorithm. Actually, N.Megiddo and M.Shub showed that the affine scaling algorithm might visit the neighborhoods of all the vertices of the Klee-Minty cube when a starting point is pushed to the boundary."**

(5) In practice, VMF reported

| | **# of iterations** |
|---|---|
| **Simplex** | $0.7159m^{0.9522}n^{0.3109}$ |
| **Affine Scaling** | $7.3385m^{-0.0187}n^{0.1694}$ |

(6) It may lose primal feasibility due to machine accuracy (Phase - I again).

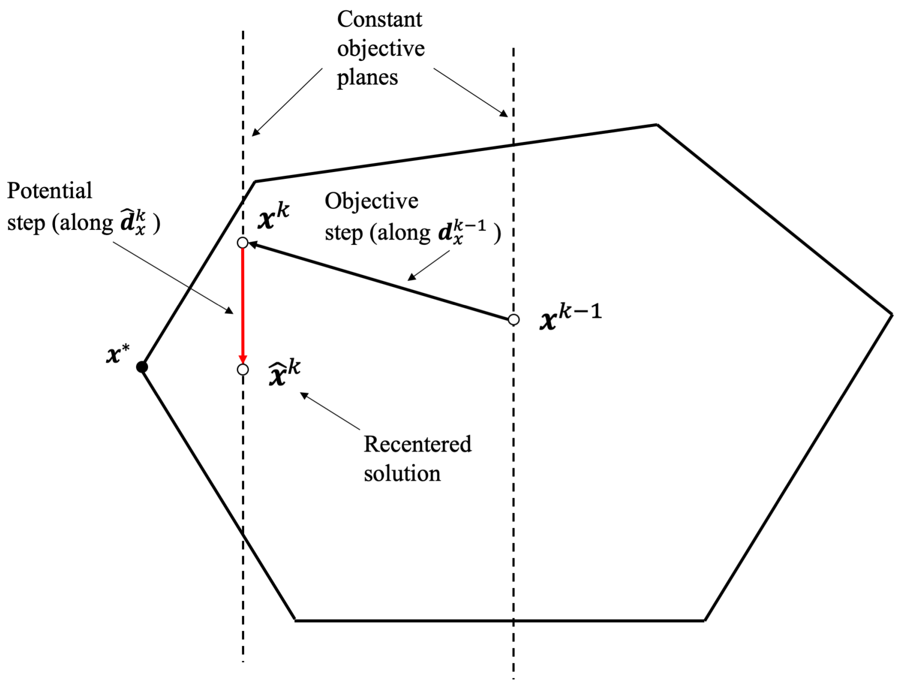
(7) Maybe sensitive to primal degeneracy.

● Improving performance - potential push

To avoid being trapped by the boundary behavior, a recentering method called *potential push* is introduced. The idea is to push a current solution $\boldsymbol{x}^k$ to a new interior solution $\hat{\boldsymbol{x}}^k$ which is away from the positivity walls but without increasing its objective value. **Then continue the iterations from $\hat{\boldsymbol{x}}^k$.**

Idea(**Potential push method**):

-Stay away from the boundary by **adding a potential push**. The mechanism is showed in the following figure.




In the figure, we move from $\boldsymbol{x}^{k-1}$ to a new solution $\boldsymbol{x}^k$ along the direction $\boldsymbol{d}_x^{k-1}$ provided by the primal affine scaling algorithm. Then we recenter $\boldsymbol{x}^k$ to $\hat{\boldsymbol{x}}^k$ by a "potential push" along the direction $\hat{\boldsymbol{d}}_x^k$ such that $\boldsymbol{x}^k$ and $\hat{\boldsymbol{x}}^k$ have the same objective value but $\hat{\boldsymbol{x}}^k$ is away from the boundary.

To achieve this goal, we define a *potential function* $p(\boldsymbol{x})$, for each $\boldsymbol{x} > \boldsymbol{0}$:

$$p(\boldsymbol{x}) = -\sum_{j=1}^{n} \log_e x_j$$

The value of the potential function $p(\boldsymbol{x})$ **becomes larger when $\boldsymbol{x}$ is closer to a positivity wall $x_j = 0$**. Hence it creates a force to **"push" $\boldsymbol{x}$ away from too close an approach to a boundary** by minimizing $p(\boldsymbol{x})$. With the potential function, we focus on solving the following "potential push" problem:

$$\text{(potential push)} \begin{cases} \text{Minimize} & p(\boldsymbol{x}) \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} > \boldsymbol{0} \\ & \boldsymbol{c}^T\boldsymbol{x} = \boldsymbol{c}^T\boldsymbol{x}^k \end{cases}$$

● Improving performance - logarithmic barrier function

Another way to stay away from the positivity walls is to incorporate a barrier function, with extremely high values along the boundaries

$$\{\boldsymbol{x} \in R^n \,|\, x_j = 0, \text{ for some } 1 \le j \le n\}$$

into the original objective function. Minimizing this new objective function will automatically push a solution away from the positivity walls. The logarithmic barrier method considers the following nonlinear optimization problem:

$$\begin{aligned} &\text{Minimize} \quad F_\mu(\boldsymbol{x}) = \boldsymbol{c}^T\boldsymbol{x} - \mu \sum_{j=1}^{n} \log_e x_j \\ &\text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b},\ \boldsymbol{x} > \boldsymbol{0} \end{aligned} \tag{7.46}$$

where $\mu > 0$ is a scalar. If $\boldsymbol{x}^*(\mu)$ is an optimal solution to problem (7.46), and if $\boldsymbol{x}^*(\mu)$ tends to a point $\boldsymbol{x}^*$ as $\mu$ approaches zero, **then it follows that $\boldsymbol{x}^*$ is an optimal solution to the original linear programming problem**. Also notice that the positivity constraint $\boldsymbol{x} > \boldsymbol{0}$ is actually **embedded in the definition** of the logarithmic function. Hence, for any fixed $\mu > 0$, the **Newton search direction $\boldsymbol{d}_\mu$** at a given feasible solution $\boldsymbol{x}$ is obtained by solving the following quadratic optimization problem:

$$\begin{aligned} &\text{Minimize} \quad \frac{1}{2}\boldsymbol{d}^T \nabla^2 F_\mu(\boldsymbol{x})\boldsymbol{d} + [\nabla F_\mu(\boldsymbol{x})]^T\boldsymbol{d} \\ &\text{subject to} \quad \boldsymbol{A}\boldsymbol{d} = \boldsymbol{0} \end{aligned}$$

where $\nabla F_\mu(\boldsymbol{x}) = \boldsymbol{c} - \mu \boldsymbol{X}^{-1}\boldsymbol{e}$ and $\nabla^2 F_\mu(\boldsymbol{x}) = \mu \boldsymbol{X}^{-2}$.

In other words, the **Newton direction** is **in the null space of matrix $A$ and it minimizes the quardratic approximation of $F_\mu(x)$**. We let $\boldsymbol{\lambda}_\mu$ denote the vector of Lagrange multipliers, then $\boldsymbol{d}_\mu$ and $\boldsymbol{\lambda}_\mu$ satisfy the following system of equations(**KKT matrix form, if you do not understand, please see the relevant textbook like Convex Optimization**):

$$\begin{pmatrix} \mu \boldsymbol{X}^{-2} & \boldsymbol{A}^T \\ \boldsymbol{A} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{d}_\mu \\ \boldsymbol{\lambda}_\mu \end{pmatrix} = - \begin{pmatrix} \boldsymbol{c} - \mu \boldsymbol{X}^{-1}\boldsymbol{e} \\ \boldsymbol{0} \end{pmatrix}$$

Letting $\boldsymbol{\phi}_\mu = \boldsymbol{X}^{-1}\mu \boldsymbol{d}_\mu$, we have

$$\begin{pmatrix} \boldsymbol{I} & \boldsymbol{X}\boldsymbol{A}^T \\ \boldsymbol{A}\boldsymbol{X} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\phi}_\mu \\ \boldsymbol{\lambda}_\mu \end{pmatrix} = - \begin{pmatrix} \boldsymbol{X}\boldsymbol{c} - \mu \boldsymbol{e} \\ \boldsymbol{0} \end{pmatrix}$$

It follows that

$$\boldsymbol{\phi}_\mu = -[\boldsymbol{I} - \boldsymbol{X}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{X}^2\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{X}](\boldsymbol{X}\boldsymbol{c} - \mu \boldsymbol{e})$$

and

$$\boldsymbol{d}_\mu = \frac{1}{\mu}X\boldsymbol{\phi}_\mu = -\frac{1}{\mu}X[\boldsymbol{I} - \boldsymbol{X}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{X}^2\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{X}](\boldsymbol{X}\boldsymbol{c} - \mu \boldsymbol{e}) \tag{7.49}$$

$$\Rightarrow \quad \boldsymbol{d}_\mu^k = \frac{1}{\mu} X_k \boldsymbol{\phi}_\mu = -\frac{1}{\mu} X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k](X_k \boldsymbol{c} - \mu \boldsymbol{e})$$

Taking the given solution to be $\boldsymbol{x} = \boldsymbol{x}^k$ and comparing $\boldsymbol{d}_\mu$ with the primal affine scaling moving direction $\boldsymbol{d}_x^k$, we see that

$$\boldsymbol{d}_\mu^k = \frac{1}{\mu} \boldsymbol{d}_x^k + X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e}$$

**The derivation in detail:**

$$\begin{aligned}
&\boldsymbol{d}_\mu^k - X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e} \\
&= -\frac{1}{\mu} X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k](X_k \boldsymbol{c} - \mu \boldsymbol{e}) - X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e} \\
&= \frac{1}{\mu} X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k](-X_k \boldsymbol{c} + \mu \boldsymbol{e}) - X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e} \\
&= \frac{1}{\mu} X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k](-X_k \boldsymbol{c}) + X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e} - X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e} \\
&= \frac{1}{\mu} X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k](-X_k \boldsymbol{c}) \\
&= \frac{1}{\mu} X_k \boldsymbol{d}_y^k \\
&= \frac{1}{\mu} \boldsymbol{d}_x^k
\end{aligned}$$

The additional component $X_k [\boldsymbol{I} - X_k \boldsymbol{A}^T (\boldsymbol{A} X_k^2 \boldsymbol{A}^T)^{-1} \boldsymbol{A} X_k] \boldsymbol{e} = X_k \boldsymbol{P}_k \boldsymbol{e}$ can be viewed as a force which pushes a solution away from the boundary. Hence some people call it a "*centering force*", and call the logarithmic barrier method a "*primal affine scaling algorithm with centering force*".

While classical barrier function theory requires that $\boldsymbol{x}^k$ solves problem (7.46) explicitly before $\mu = \mu_k$ is reduced. C.Gonzaga has pointed out that there exists $\mu_0 > 0,\ 0 < \rho < 1$, and $\alpha > 0$ so that choosing $\boldsymbol{d}_\mu^k$ by (7.49), $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha \boldsymbol{d}_\mu^k$ and $\mu_{k+1} = \rho \mu_k$ yields convergence to an optimal solution $\boldsymbol{x}^*$ to the original linear programming problem in $O(\sqrt{n}L)$ iterations. This could result in a polynomial-time affine scaling algorithm with complexity $O(n^3 L)$. A simple and elegant proof is due to C.Roos and J.-Ph.Vial, similar to the one proposed by R.Monteiro and I.Adler for the primal-dual algorithm.

- Dual affine scaling algorithm

Affine scaling method applied to the dual LP

$$(\mathrm{D}) \begin{cases} \max & \boldsymbol{b}^T \boldsymbol{w} \\ \text{s.t.} & \boldsymbol{A}^T \boldsymbol{w} + \boldsymbol{s} = \boldsymbol{c} \\ & \boldsymbol{s} \geq \boldsymbol{0} \end{cases}$$

Idea: given $(\boldsymbol{w}^k, \boldsymbol{s}^k)$ dual interior feasible, i.e.,

$$\boldsymbol{A}^T \boldsymbol{w}^k + \boldsymbol{s}^k = \boldsymbol{c}$$
$$\boldsymbol{s}^k > \boldsymbol{0}$$

Objective: find $(\boldsymbol{d}_w^k, \boldsymbol{d}_s^k)$ and step-length $\beta_k > 0$ such that

$$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k + \beta_k \boldsymbol{d}_w^k$$
$$\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \beta_k \boldsymbol{d}_s^k$$

is still dual interior feasible and $\boldsymbol{b}^T\boldsymbol{w}^{k+1} \geq \boldsymbol{b}^T\boldsymbol{w}^k$.

- Key knowledge

Dual **scaling (centering)**
Dual **feasible direction**
Dual **good direction** - increase the dual objective value
Dual **step-length**
**Primal estimate** for stopping rule

- Observation 1

Dual scaling (centering)
$\boldsymbol{w}^k \in R^m$: no scaling needed.
$\boldsymbol{s}^k > \boldsymbol{0}$: scale to $\boldsymbol{e} = (1,1,\cdots,1)^T \in R^n$

$$\boldsymbol{S}_k = \text{diag}(s_i^k) = \begin{pmatrix} s_1^k & 0 & \cdots & 0 \\ 0 & s_2^k & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_n^k \end{pmatrix}$$

$$\boldsymbol{u}^k = \boldsymbol{S}_k^{-1}\boldsymbol{s}^k \qquad \boldsymbol{d}_u^k = \boldsymbol{S}_k^{-1}\boldsymbol{d}_s^k$$
$$\boldsymbol{s}^k = \boldsymbol{S}_k\boldsymbol{u}^k \qquad \boldsymbol{d}_s^k = \boldsymbol{S}_k\boldsymbol{d}_u^k$$

- Observation 2

Dual feasibility (feasible direction)

$$\underbrace{\boldsymbol{A}^T\boldsymbol{w}^{k+1} + \boldsymbol{s}^{k+1}}_{\boldsymbol{c}} = \boldsymbol{A}^T(\boldsymbol{w}^k + \beta_k\boldsymbol{d}_w^k) + (\boldsymbol{s}^k + \beta_k\boldsymbol{d}_s^k)$$
$$= \underbrace{(\boldsymbol{A}^T\boldsymbol{w}^k + \boldsymbol{s}^k)}_{\boldsymbol{c}} + \underbrace{\beta_k}_{>0}(\boldsymbol{A}^T\boldsymbol{d}_w^k + \boldsymbol{d}_s^k)$$

$\Rightarrow \boldsymbol{A}^T\boldsymbol{d}_w^k + \boldsymbol{d}_s^k = \boldsymbol{0}$ is required!

$$\Longleftrightarrow \boldsymbol{S}_k^{-1}\boldsymbol{A}^T\boldsymbol{d}_w^k + \underbrace{\boldsymbol{S}_k^{-1}\boldsymbol{d}_s^k}_{\boldsymbol{d}_u^k} = \boldsymbol{0}$$

(NOTE: $\boldsymbol{S}_k^{-1}\boldsymbol{A}^T$ is not a square matrix thus it does not have inverse matrix!)

$$\Longleftrightarrow \boldsymbol{A}\boldsymbol{S}_k^{-1}(\boldsymbol{S}_k^{-1}\boldsymbol{A}^T\boldsymbol{d}_w^k + \boldsymbol{d}_u^k) = \boldsymbol{0}$$
$$\Longleftrightarrow (\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)\boldsymbol{d}_w^k + \boldsymbol{A}\boldsymbol{S}_k^{-1}\boldsymbol{d}_u^k = \boldsymbol{0}$$
$$\Longleftrightarrow \boldsymbol{d}_w^k = -\underbrace{(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{S}_k^{-1}}_{\boldsymbol{Q}_k}\boldsymbol{d}_u^k$$

● Observation 3

Increase dual objective function (good direction)

$$\boldsymbol{b}^T\boldsymbol{w}^{k+1} = \boldsymbol{b}^T\boldsymbol{w}^k + \beta_k \boldsymbol{b}^T\boldsymbol{d}_w^k \geq \boldsymbol{b}^T\boldsymbol{w}^k$$

Thus

$$\boldsymbol{b}^T\boldsymbol{d}_w^k = -\boldsymbol{b}^T\boldsymbol{Q}_k\boldsymbol{d}_u^k \geq \boldsymbol{0}$$

Choose $\boldsymbol{d}_u^k = -\boldsymbol{Q}_k^T\boldsymbol{b}$, then

$$\boldsymbol{b}^T\boldsymbol{d}_w^k = \boldsymbol{b}^T\boldsymbol{Q}_k\boldsymbol{Q}_k^T\boldsymbol{b} = \|\boldsymbol{Q}_k^T\boldsymbol{b}\|^2 \geq 0$$

According to $\boldsymbol{d}_u^k = -\boldsymbol{Q}_k^T\boldsymbol{b}$ and $\boldsymbol{d}_w^k = -\boldsymbol{Q}_k\boldsymbol{d}_u^k$,

$$\begin{aligned}
\boldsymbol{d}_w^k &= -\boldsymbol{Q}_k\boldsymbol{d}_u^k = \boldsymbol{Q}_k\boldsymbol{Q}_k^T\boldsymbol{b} \\
\Rightarrow \quad &= \underbrace{(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{S}_k^{-1}}_{\boldsymbol{Q}_k}\ \underbrace{\boldsymbol{S}_k^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}}_{\boldsymbol{Q}_k^T}\boldsymbol{b} \\
&= (\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}\boldsymbol{b}
\end{aligned}$$

According to $\boldsymbol{A}^T\boldsymbol{d}_w^k + \boldsymbol{d}_s^k = \boldsymbol{0}$,

$\Rightarrow \boldsymbol{d}_s^k = -\boldsymbol{A}^T\boldsymbol{d}_w^k = -\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A})^{-1}\boldsymbol{b}$

● Observation 4

Dual step-length

$$\boldsymbol{s}^{k+1} = \underbrace{\boldsymbol{s}^k}_{>\boldsymbol{0}} + \beta_k\boldsymbol{d}_s^k > \boldsymbol{0}$$

(i) $\boldsymbol{d}_s^k = \boldsymbol{0}$, problem (D) has a constant objective value and $(\boldsymbol{w}^k, \boldsymbol{s}^k)$ are dual optimal.

(ii) $\boldsymbol{d}_s^k \geq \boldsymbol{0}$ (but $\neq \boldsymbol{0}$), $\beta_k \in (0, +\infty)$, problem (D) is unbounded above.

(iii) some $(\boldsymbol{d}_s^k)_i < 0$

$$\beta_k = \min_i\{\frac{\alpha s_i^k}{-(\boldsymbol{d}_s^k)_i} \mid (\boldsymbol{d}_s^k)_i < 0\} \quad \text{where} \quad 0 < \alpha < 1.$$

● Observation 5

Primal estimate

We define $\boldsymbol{x}^k \stackrel{\text{def}}{=} -\boldsymbol{S}_k^{-2}\boldsymbol{d}_s^k$

then

$$\begin{aligned}
\boldsymbol{A}\boldsymbol{x}^k &= -\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{d}_s^k = -\boldsymbol{A}\boldsymbol{S}_k^{-2}[-\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}\boldsymbol{b}] \\
&= \boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}\boldsymbol{b} \\
&= \boldsymbol{b}
\end{aligned}$$

Hence $\boldsymbol{x}^k$ is a primal estimate, once $\boldsymbol{x}^k \geq \boldsymbol{0}$, then $\boldsymbol{x}^k$ is primal feasible.

Furthermore, if $\boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{w} = 0$, then $\boldsymbol{x}^k \leftarrow \boldsymbol{x}^*$, $\boldsymbol{w}^k \leftarrow \boldsymbol{w}^*$, $\boldsymbol{s}^k \leftarrow \boldsymbol{s}^*$.

Key steps of dual affine scaling algorithm:

Step 1: Set $k = 0$, $\epsilon > 0$(small enough) and find $(\boldsymbol{w}^0, \boldsymbol{s}^0)$, s.t. $\boldsymbol{A}^T\boldsymbol{w}^0 + \boldsymbol{s}^0 = \boldsymbol{c}, \boldsymbol{s}^0 > \boldsymbol{0}$.

Step 2: Set $\boldsymbol{S}_k = \text{diag}(s_i^k)$.

Compute $\boldsymbol{d}_w^k = (\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A}^T)^{-1}\boldsymbol{b},\ \boldsymbol{d}_s^k = -\boldsymbol{A}^T\boldsymbol{d}_w^k$.

Step 3: If $\boldsymbol{d}_s^k = \boldsymbol{0}$, STOP! $\boldsymbol{w}^k \leftarrow \boldsymbol{w}^*, \boldsymbol{s}^k \leftarrow \boldsymbol{s}^*$.

If $\boldsymbol{d}_s^k \geq \boldsymbol{0}$ (but $\neq \boldsymbol{0}$), STOP! (D) is unbounded above.

Otherwise,

Step 4: Compute

$\boldsymbol{x}^k = -\boldsymbol{S}_k^{-2}\boldsymbol{d}_s^k$

If $\boldsymbol{x}^k \geq \boldsymbol{0}$ and $\boldsymbol{c}^T\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{w} \leq \epsilon$, STOP! $\boldsymbol{x}^k \leftarrow \boldsymbol{x}^*, \boldsymbol{w}^k \leftarrow \boldsymbol{w}^*, \boldsymbol{s}^k \leftarrow \boldsymbol{s}^*$.

Otherwise,

Step 5: Compute

$$\beta_k = \min_i\{\frac{\alpha s_i^k}{-(\boldsymbol{d}_s^k)_i} \mid (\boldsymbol{d}_s^k)_i < 0\} \quad \text{where } 0 < \alpha < 1.$$

Step 6: Update

$\boldsymbol{w}^{k+1} = \boldsymbol{w}^k + \beta_k\boldsymbol{d}_w^k$

$\boldsymbol{s}^{k+1} = \boldsymbol{s}^k + \beta_k\boldsymbol{d}_s^k$

Set $k \leftarrow k+1$, go to Step 2.

● Find an initial interior feasible solution

Find $(\boldsymbol{w}^0, \boldsymbol{s}^0)$, s.t.

$$\boldsymbol{A}^T\boldsymbol{w}^0 + \boldsymbol{s}^0 = \boldsymbol{c}$$
$$\boldsymbol{s}^0 > \boldsymbol{0}$$

If $\boldsymbol{c} > \boldsymbol{0}$, then $\boldsymbol{w}^0 = \boldsymbol{0}, \boldsymbol{s}^0 = \boldsymbol{c}$ will do.

● Big-M method:

Define $\boldsymbol{p} \in R^n, p_i = \begin{cases} 1, & \text{if } c_i \leq 0 \\ 0, & \text{if } c_i > 0 \end{cases}$

Consider, for a large $M > 0$,

$$(\text{Big - M}) \begin{cases} \max & \boldsymbol{b}^T\boldsymbol{w} + Mw^a \\ \text{s.t.} & \boldsymbol{A}^T\boldsymbol{w} + \boldsymbol{p}w^a + \boldsymbol{s} = \boldsymbol{c} \\ & \boldsymbol{s} \geq \boldsymbol{0} \end{cases}$$

Properties of (Big - M) problem:

(a) (Big - M) is a standard LP with $n$ constraints and $m + 1 + n$ variables.

(b) Define $\bar{c} = \max_i |c_i|$ and $\theta > 1$, then

$$\boldsymbol{w} = \boldsymbol{0}, w^a = -\theta\bar{c},\ \boldsymbol{s} = \boldsymbol{c} + \theta\bar{c}\boldsymbol{p} > \boldsymbol{0}$$

is an initial interior feasible solution for problem (D).

(c) $(w^a)^0 = -\theta\bar{c} < 0$.

Since $M > 0$ is large, $(w^a)^k \to 0$ as $k \to +\infty$.

If $(w^a)^k$ does not approach or cross zero, then problem (D) is infeasible.

● Two-phase method

$$(\text{D}) \begin{cases} \max & \boldsymbol{b}^T \boldsymbol{w} \\ \text{s.t.} & \boldsymbol{A}^T \boldsymbol{w} + \boldsymbol{s} = \boldsymbol{c} \\ & \boldsymbol{s} \geq \boldsymbol{0} \end{cases}$$

Choose any $\boldsymbol{w}^0$ and $\boldsymbol{s}^0 > \boldsymbol{0}$, calculate $\boldsymbol{v} = \boldsymbol{c} - \boldsymbol{A}^T \boldsymbol{x}^0 - \boldsymbol{s}^0$.
If $\boldsymbol{v} = \boldsymbol{0}$, then $(\boldsymbol{w}^0, \boldsymbol{s}^0)$ is dual interior feasible.

Otherwise, consider

$$(\text{Phase - I}) \begin{cases} \min & u \\ \text{s.t.} & \boldsymbol{A}^T \boldsymbol{w} + \boldsymbol{s} + \boldsymbol{v} u = \boldsymbol{c} \\ & \boldsymbol{s} \geq \boldsymbol{0},\ u \geq 0 \end{cases}$$

**Properties of (Phase - I) problem:**

(1) (Phase - I) is a standard form LP with $m + n + 1$ variables and $n$ constraints.

(2) $\begin{pmatrix} \boldsymbol{w}^0 \\ \boldsymbol{s}^0 \\ u^0 \end{pmatrix} = \begin{pmatrix} \boldsymbol{w}^0 \\ \boldsymbol{s}^0 \\ 1 \end{pmatrix}$ is dual interior feasible for (Phase - I).

(3) (Phase - I) is bounded below by 0.

(4) Apply dual-affine scaling to (Phase - I) will generate $\begin{pmatrix} \boldsymbol{w}^* \\ \boldsymbol{s}^* \\ u^* \end{pmatrix}$ for (Phase - I).

(5) If $u^* > 0$, (D) is infeasible.
Otherwise, $(\boldsymbol{w}^*, \boldsymbol{s}^*)$ is an initial feasible solution for (D).

● Performance of dual affine scaling

**No polynomial-time poof!**

Computational bottleneck $(\boldsymbol{A}\boldsymbol{S}_k^{-2}\boldsymbol{A})^{-1}$.

Less sensitive to primal degeneracy and numerical errors but **sensitive to dual degeneracy**.

Improve dual objective value very fast, but **attains primal feasibility slowly**.

● Improving performance

Logarithmic barrier function method

Similar to that of the primal affine scaling algorithm, we can incorporate a barrier function, with extremely high values along the boundaries $\{(\boldsymbol{w}; \boldsymbol{s}) \mid s_j = 0, \text{ for some } 1 \leq j \leq n\}$, into the original objective function. Now, consider the following the nonlinear programming problem:

$$\begin{aligned} \max \quad & F_\mu(\boldsymbol{w}, \mu) = \boldsymbol{b}^T \boldsymbol{w} + \mu \sum_{j=1}^{n} \log_e(c_j - \boldsymbol{A}_j^T \boldsymbol{w}) \\ \text{s.t.} \quad & \boldsymbol{A}^T \boldsymbol{w} < \boldsymbol{c} \end{aligned} \tag{7.72}$$

where $\mu > 0$ is a scalar and $\boldsymbol{A}_j^T$ is the transpose of the $j$th column vector of matrix $\boldsymbol{A}$. Note that if $\boldsymbol{w}^*(\mu)$ is an optimal solution to problem (7.72), and if $\boldsymbol{w}^*(\mu)$ tends to a point $\boldsymbol{w}^*$as $\mu$ approaches zero, then it follows that $\boldsymbol{w}^*$ is an optimal solution to the original dual linear programming problem.

The Lagrangian of problem (7.72) becomes

$$L(\boldsymbol{w}, \lambda) = \boldsymbol{b}^T \boldsymbol{w} + \mu \sum_{j=1}^{n} \log_e(c_j - \boldsymbol{A}_j^T \boldsymbol{w}) + \boldsymbol{\lambda}^T(\boldsymbol{c} - \boldsymbol{A}^T \boldsymbol{w})$$

where $\boldsymbol{\lambda}$ is a vector of Lagarangian multipliers. Since $c_j - \boldsymbol{A}_j^T \boldsymbol{w} > 0$, the complementary slackness condition requires that $\boldsymbol{\lambda} = \boldsymbol{0}$, and the associated K-K-T conditons become

$$\boldsymbol{b} - \mu \boldsymbol{A} \boldsymbol{S}^{-1} \boldsymbol{e} = \boldsymbol{0} \text{ (stationary \& } \boldsymbol{\lambda} = \boldsymbol{0}\text{) and } \boldsymbol{s} > \boldsymbol{0}$$

Assuming that $\boldsymbol{w}^k$ and $\boldsymbol{s}^k = \boldsymbol{c} - \boldsymbol{A}^T \boldsymbol{w}^k > \boldsymbol{0}$ form a current dual feasible solution, we take one Newton step of the K-K-T conditions. This results in a moving direction

$$\Delta \boldsymbol{w} = \frac{1}{\mu} \underbrace{(\boldsymbol{A} \boldsymbol{S}_k^{-2} \boldsymbol{A}^T)^{-1} \boldsymbol{b}}_{\boldsymbol{d}_w^k} - (\boldsymbol{A} \boldsymbol{S}_k^{-2} \boldsymbol{A}^T)^{-1} \boldsymbol{A} \boldsymbol{S}_k^{-1} \boldsymbol{e}$$

We see that $-(\boldsymbol{A} \boldsymbol{S}_k^{-2} \boldsymbol{A}^T)^{-1} \boldsymbol{A} \boldsymbol{S}_k^{-1} \boldsymbol{e}$ is an additional term in the logarithmic barrier method to push a solution away from the boundary. Therefore, sometimes the logarithmic barrier function method is called *dual affine scaling with centering force*.

$\boldsymbol{b} - \mu \boldsymbol{A} \boldsymbol{S}^{-1} \boldsymbol{e} = \boldsymbol{0}$

The derivation in detail:

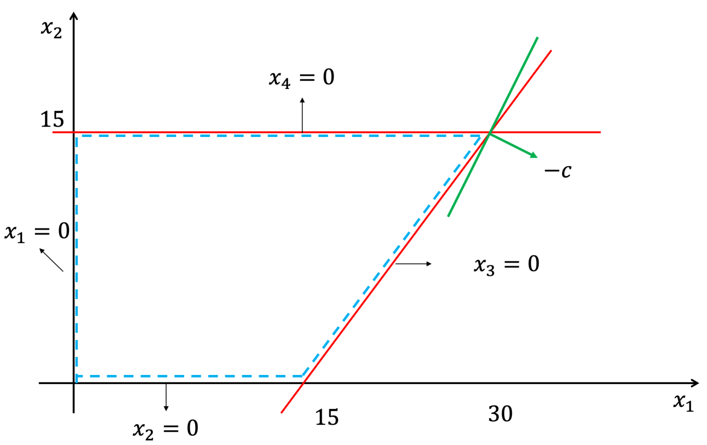
$$\boldsymbol{\lambda} = \boldsymbol{0} \Rightarrow L(\boldsymbol{w}, \lambda = \boldsymbol{0}) = \boldsymbol{b}^T \boldsymbol{w} + \mu \sum_{j=1}^{n} \log_e(c_j - \boldsymbol{A}_j^T \boldsymbol{w})$$

$$\frac{\mathrm{d}L(\boldsymbol{w}, \boldsymbol{\lambda} = \boldsymbol{0})}{\mathrm{d}w_i} = b_i + \mu \sum_{j=1}^{n} \frac{-a_{ij}}{c_j - \boldsymbol{A}_j^T \boldsymbol{w}} = b_i - \mu \sum_{j=1}^{n} \frac{a_{ij}}{c_j - \boldsymbol{A}_j^T \boldsymbol{w}} = b_i - \mu \sum_{j=1}^{n} \frac{a_{ij}}{s_j}$$

Appendix(example for primal affine scaling algorithm)

- Example:

$$\begin{aligned} \min \quad & -2x_1 + x_2 \\ \text{s.t.} \quad & x_1 - x_2 \le 15 \\ & x_2 \le 15 \\ & x_1, x_2 \ge 0 \end{aligned}$$



Reformulate to standard form:

$$\begin{aligned}\min \quad & -2x_1 + x_2 + 0x_3 + 0x_4 \\ \text{s.t.} \quad & x_1 - x_2 + x_3 \quad = 15 \\ & \quad x_2 + \quad x_4 = 15 \\ & x_1, x_2, x_3, x_4 \geq 0\end{aligned} \qquad \boldsymbol{x}^0 = \begin{bmatrix}10\\2\\7\\13\end{bmatrix} \text{ is feasible, } \boldsymbol{X}_0 = \begin{pmatrix}10 & 0 & 0 & 0\\0 & 2 & 0 & 0\\0 & 0 & 7 & 0\\0 & 0 & 0 & 13\end{pmatrix}$$

$$\boldsymbol{A} = \begin{bmatrix}1 & -1 & 1 & 0\\0 & 1 & 0 & 1\end{bmatrix} \qquad \boldsymbol{b} = \begin{bmatrix}15\\15\end{bmatrix} \qquad \boldsymbol{c} = \begin{bmatrix}-2\\1\\0\\0\end{bmatrix}$$

Hence

$$\boldsymbol{X}_0 = \begin{pmatrix}10 & 0 & 0 & 0\\0 & 2 & 0 & 0\\0 & 0 & 7 & 0\\0 & 0 & 0 & 13\end{pmatrix} \text{ and } \boldsymbol{w}^0 = (\boldsymbol{A}\boldsymbol{X}_0^2\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{X}_0^2\boldsymbol{c} = [-1.33353, -0.00771]^T$$

Moreover,

$$\boldsymbol{r}^0 = \boldsymbol{c} - \boldsymbol{A}^T\boldsymbol{w}^0 = [-0.66647, -0.32582, 1.33353, -0.00771]^T$$

Since some components of $\boldsymbol{r}^0$ are nagative and $\boldsymbol{e}^T\boldsymbol{X}_0\boldsymbol{r}^0 = 2.1187$, we know that the current solution is nonoptimal. Therefore we proceed to synthesize the direction of translation with

$$\boldsymbol{d}_y^0 = -\boldsymbol{X}_0\boldsymbol{r}^0 = [6.6647, 0.6516, -9.3347, 0.1002]^T$$

Suppose that $\alpha = 0.99$ is chosen, the step-length

$$\alpha_0 = \frac{0.99}{9.3347} = 0.1061$$

Therefore the new solution is

$$\boldsymbol{x}^1 = \boldsymbol{x}^0 + \alpha_0\boldsymbol{X}_0\boldsymbol{d}_y^0 = [17.07119, 2.13828, 0.06709, 12.86172]^T$$

Notice that the objective function values has been improved from $-18$ to $-32.0041$. The reader may continue the iterations further and verify that the iterative process converges to the optimal solution $\boldsymbol{x}^* = [30, 15, 0, 0]^T$ with optimal value $-45$.