

Chapter 7

Interior Point Methods for Linear and Conic Programming

Historical Background

- 1945 **Simplex algorithm** by G. Dantzig
- 1968 **Barrier method** by A. Fiacco and G. Mc Cormick for nonlinear convex programming
- 1970 Simplex algorithm is proven to be exponential in the worst case
- 1978 L. Khachiyan applies the **ellipsoid method** to prove that any conex program (including LP) can be solved in polynomial time.
The algorithm has **poor practical performances**.
- 1984 N. Karmarkar announces that he can solve linear programs more quickly than with the simplex algorithm using an **interior point method**. (however partially true only)
- 1994 Y. Nesterov and A. Nemirovski publish a complete treatment of semidefinite, conic quadratic and linear problems using interior point methods.

The idea of a barrier function

Starting from

$$\begin{aligned}(\mathcal{P}) \equiv \min f(x) \\ \text{s.t. } g(x) \geq 0\end{aligned}$$

A barrier

A **barrier** is a differentiable function $\phi : \mathbb{R}_+ \mapsto \mathbb{R}$ such that ϕ is defined on the positive numbers and

$$\lim_{x \rightarrow 0^+} \phi(x) = +\infty.$$

Now we can solve the **constrained** problem as a **family** of **unconstrained** problems

$$(\mathcal{P})_\mu \equiv \min f(x) - \mu\phi(g(x)).$$

Idea of barrier methods

Let $x(\mu)$ be an **optimal solution** of $(\mathcal{P})_\mu$.

An optimal solution of (\mathcal{P}) might be $\lim_{\mu \rightarrow 0} x(\mu)$.

The idea of a barrier function

Starting from

$$\begin{aligned}(\mathcal{P}) \equiv \min f(x) \\ \text{s.t. } g(x) \geq 0\end{aligned}$$

A barrier

A **barrier** is a differentiable function $\phi : \mathbb{R}_+ \mapsto \mathbb{R}$ such that ϕ is defined on the positive numbers and

$$\lim_{x \rightarrow 0^+} \phi(x) = +\infty.$$

Now we can solve the **constrained** problem as a **family** of **unconstrained** problems

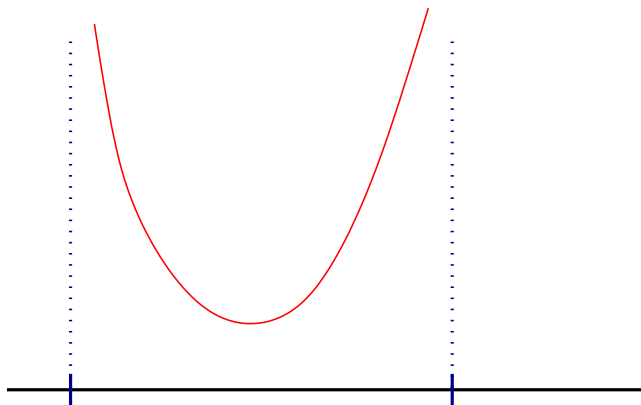
$$(\mathcal{P})_\mu \equiv \min f(x) - \mu\phi(g(x)).$$

Idea of barrier methods

Let $x(\mu)$ be an **optimal solution** of $(\mathcal{P})_\mu$.

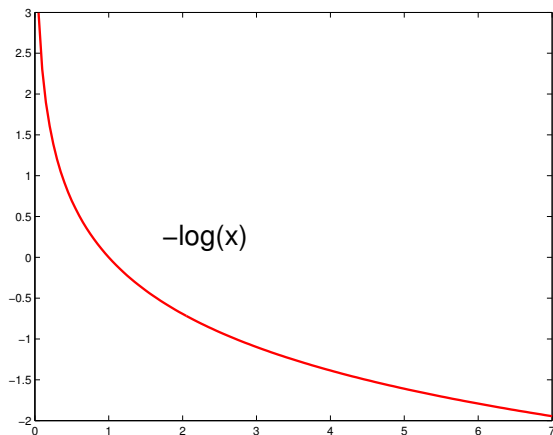
An optimal solution of (\mathcal{P}) might be $\lim_{\mu \rightarrow 0} x(\mu)$.

The idea of a barrier function



A barrier for a constraint on an interval

The idea of a barrier function



A logarithmic barrier for the $x \geq 0$ constraint

The central path

We will use the **logarithmic barrier** to encode the $x \geq 0$ constraints both in the **primal** and in the **dual** problem.

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & p^T b \\ \text{s.t.} \quad & A^T p + s = c \\ & s \geq 0 \end{aligned}$$

The modified problems become

$$\begin{aligned} \min \quad & c^T x - \mu \sum_{i=1}^n \log(x_i) \\ \text{s.t.} \quad & Ax = b \\ & x > 0 \end{aligned}$$

$$\begin{aligned} \max \quad & p^T b + \mu \sum_{j=1}^m \log(s_j) \\ \text{s.t.} \quad & A^T p + s = c \\ & s > 0 \end{aligned}$$

For fixed μ , denote the optimal solutions to the primal and the dual $(x(\mu))$ and $(p(\mu), s(\mu))$ respectively.

The central path

We will use the **logarithmic barrier** to encode the $x \geq 0$ constraints both in the **primal** and in the **dual** problem.

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & p^T b \\ \text{s.t.} \quad & A^T p + s = c \\ & s \geq 0 \end{aligned}$$

The modified problems become

$$\begin{aligned} \min \quad & c^T x - \mu \sum_{i=1}^n \log(x_i) \\ \text{s.t.} \quad & Ax = b \\ & x > 0 \end{aligned}$$

$$\begin{aligned} \max \quad & p^T b + \mu \sum_{j=1}^m \log(s_j) \\ \text{s.t.} \quad & A^T p + s = c \\ & s > 0 \end{aligned}$$

For fixed μ , denote the optimal solutions to the primal and the dual $(x(\mu))$ and $(p(\mu), s(\mu))$ respectively.

The central path

We will use the **logarithmic barrier** to encode the $x \geq 0$ constraints both in the **primal** and in the **dual** problem.

$$\begin{array}{ll} \min c^T x & \max p^T b \\ \text{s.t. } Ax = b & \text{s.t. } A^T p + s = c \\ x \geq 0 & s \geq 0 \end{array}$$

The modified problems become

$$\begin{array}{ll} \min c^T x - \mu \sum_{i=1}^n \log(x_i) & \max p^T b + \mu \sum_{j=1}^m \log(s_j) \\ \text{s.t. } Ax = b & \text{s.t. } A^T p + s = c \\ x > 0 & s > 0 \end{array}$$

For fixed μ , denote the optimal solutions to the primal and the dual $(x(\mu))$ and $(p(\mu), s(\mu))$ respectively.

The central path

Properties of the solutions for a given μ

- The sequences $(x(\mu))$ and $p(\mu)$ are called the (primal or dual) **central path**
- The sequence $(c^T x(\mu))$ is **nonincreasing** when μ goes to 0.
The sequence $(b^T p(\mu))$ is **nondecreasing** when μ goes to 0.
- The **duality gap** $c^T x(\mu) - b^T p(\mu) = n\mu$
 μ is called the **duality measure**
If (x, p, s) are **not** on the central path, an **estimated duality measure** is
 $(c^T x(\mu) - b^T p(\mu))/n$

$$x^* = \lim_{\mu \rightarrow 0} x(\mu)$$

$$(y^*, s^*) = \lim_{\mu \rightarrow 0} (y(\mu), s(\mu))$$

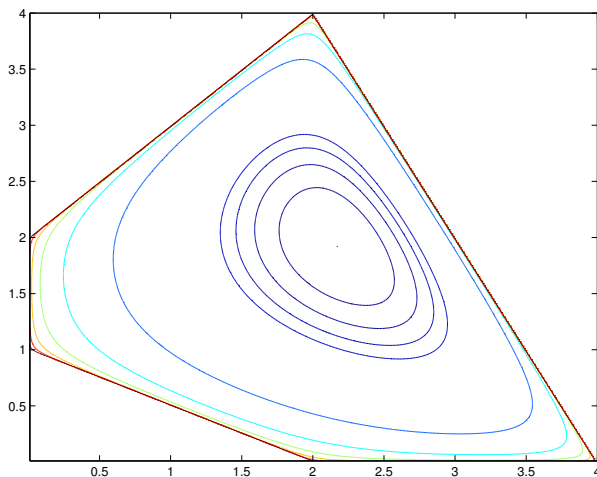
are **optimal** solutions of the primal and the dual respectively.

- The solutions (x^*) and (y^*, s^*) satisfy **strict complementarity slackness** i.e.
 $x_i^* + s_i^* > 0$.
- $(x(\mu), p(\mu), s(\mu))$ satisfy the modified KKT conditions

$$Ax = b, \quad A^T p + s = c, \quad x_i s_i = \mu.$$

The central path

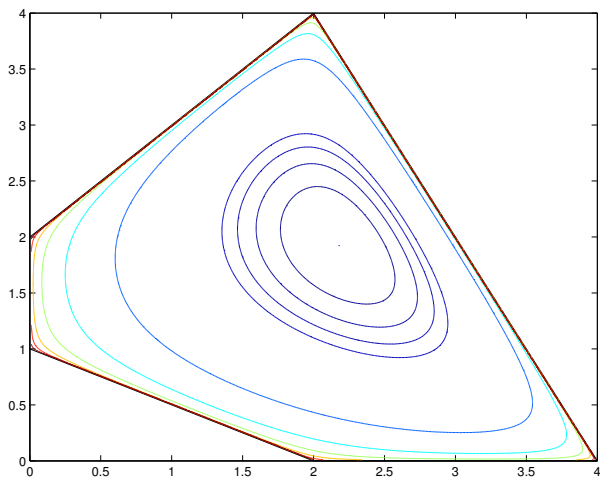
Evolution of the central path when μ decreases to 0.



$\mu = 400$

The central path

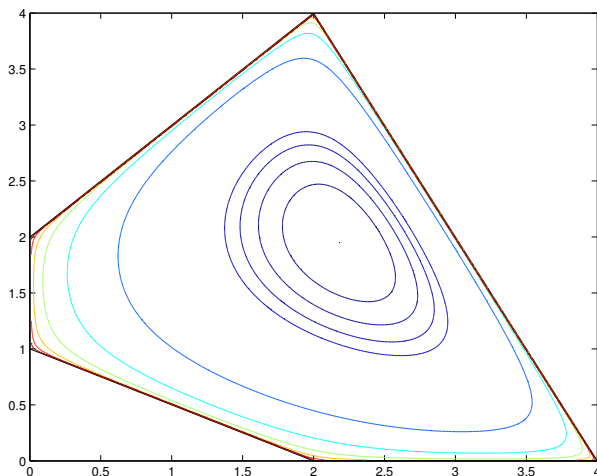
Evolution of the central path when μ decreases to 0.



$\mu = 100$

The central path

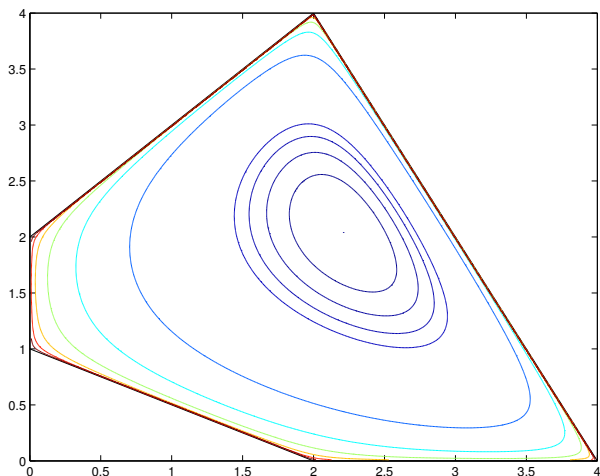
Evolution of the central path when μ decreases to 0.



$\mu = 25$

The central path

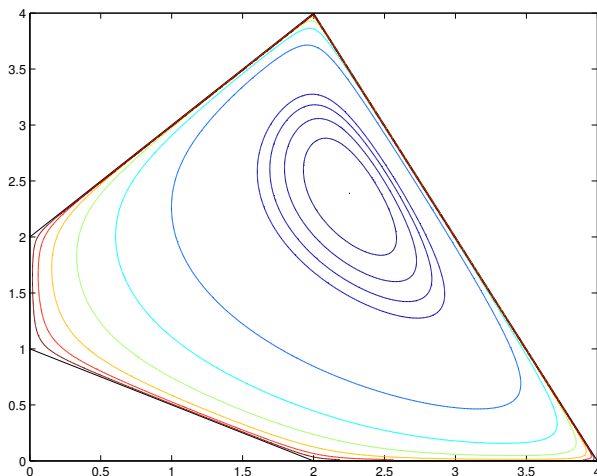
Evolution of the central path when μ decreases to 0.



$\mu = 6.4$

The central path

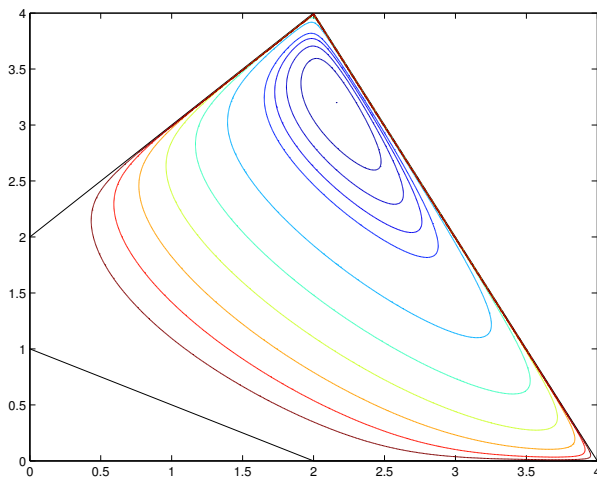
Evolution of the central path when μ decreases to 0.



$\mu = 1.6$

The central path

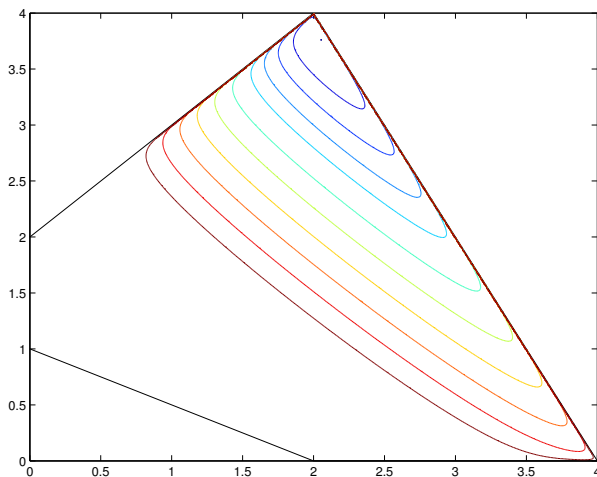
Evolution of the central path when μ decreases to 0.



$\mu = 0.4$

The central path

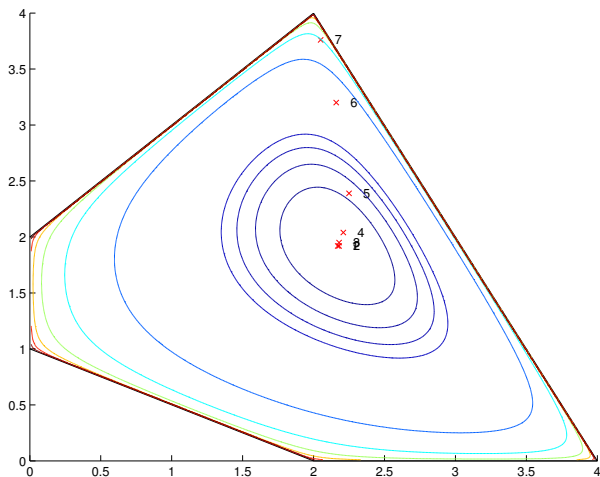
Evolution of the central path when μ decreases to 0.



$\mu = 0.1$

The central path

Evolution of the central path when μ decreases to 0.



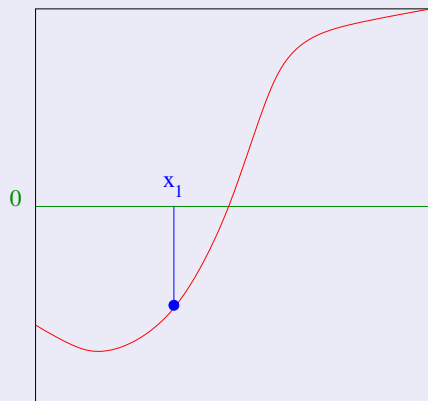
The central path

The Newton Method

Newton method to find a zero of a function

Iterative method to find a zero of a nonlinear function $f(x) = 0$.

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

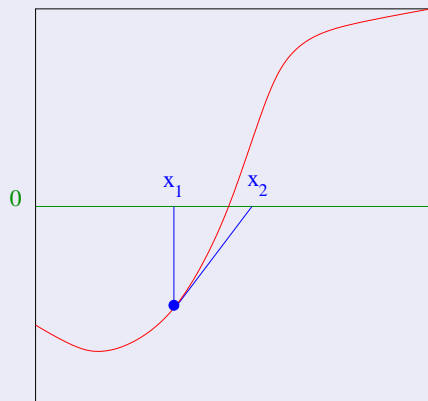


The Newton Method

Newton method to find a zero of a function

Iterative method to find a zero of a nonlinear function $f(x) = 0$.

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

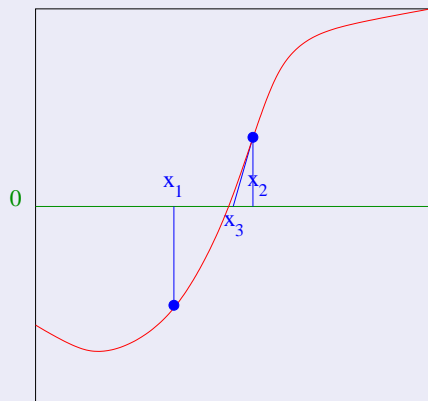


The Newton Method

Newton method to find a zero of a function

Iterative method to find a zero of a nonlinear function $f(x) = 0$.

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$



The Newton Method

Can be generalized to find the zero of a **system of nonlinear equations**

The Newton method for the systems

To solve $F(\underline{x}) = \underline{0}$,

$$\underline{x}_{k+1} = \underline{x}_k - \left(\frac{\partial F(\underline{x}_k)}{\partial \underline{x}} \right)^{(-1)} F(\underline{x}_k),$$

where $\frac{\partial F(\underline{x}_k)}{\partial \underline{x}}$ is the **Jacobian** of F .

Each step

We update $\underline{x}_{k+1} = \underline{x}_k - \underline{\Delta}_k$ where

$$\begin{pmatrix} \frac{\partial F_1(\underline{x}_k)}{\partial x_1} & \dots & \frac{\partial F_1(\underline{x}_k)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_n(\underline{x}_k)}{\partial x_1} & \dots & \frac{\partial F_n(\underline{x}_k)}{\partial x_n} \end{pmatrix} \begin{pmatrix} \Delta_{k,1} \\ \vdots \\ \Delta_{k,n} \end{pmatrix} = \begin{pmatrix} F_1(\underline{x}_k) \\ \vdots \\ F_n(\underline{x}_k) \end{pmatrix}.$$

The Newton Method

Can be generalized to find the zero of a **system of nonlinear equations**

The Newton method for the systems

To solve $F(\underline{x}) = \underline{0}$,

$$\underline{x}_{k+1} = \underline{x}_k - \left(\frac{\partial F(\underline{x}_k)}{\partial \underline{x}} \right)^{(-1)} F(\underline{x}_k),$$

where $\frac{\partial F(\underline{x}_k)}{\partial \underline{x}}$ is the **Jacobian** of F .

Each step

We update $\underline{x}_{k+1} = \underline{x}_k - \underline{\Delta}_k$ where

$$\begin{pmatrix} \frac{\partial F_1(\underline{x}_k)}{\partial x_1} & \dots & \frac{\partial F_1(\underline{x}_k)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_n(\underline{x}_k)}{\partial x_1} & \dots & \frac{\partial F_n(\underline{x}_k)}{\partial x_n} \end{pmatrix} \begin{pmatrix} \Delta_{k,1} \\ \vdots \\ \Delta_{k,n} \end{pmatrix} = \begin{pmatrix} F_1(\underline{x}_k) \\ \vdots \\ F_n(\underline{x}_k) \end{pmatrix}.$$

The Newton Method for optimization

Minimizing a univariate function with the Newton Method

Let $f : \mathbb{R} \mapsto \mathbb{R}$. The iterative process tries to find a zero of the **derivative**. If f is **convex**, the only zero of derivative corresponds to the **unique minimizer** of f .

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}.$$

Minimizing a multivariate function with the Newton Method

Let $F : \mathbb{R}^n \mapsto \mathbb{R}$, $\nabla F = \left(\frac{\partial F}{\partial x_i} \right) : \mathbb{R}^n \mapsto \mathbb{R}^n$ be the **gradient** of F and $\mathcal{H} = \left(\frac{\partial^2 F}{\partial x_i^2} \right)$ be the **Hessian** of F .

$$x^{[k+1]} = x^{[k]} - \mathcal{H}(x^{[k]})^{-1} \nabla F(x^{[k]}).$$

The Newton Method for optimization

Minimizing a univariate function with the Newton Method

Let $f : \mathbb{R} \mapsto \mathbb{R}$. The iterative process tries to find a zero of the **derivative**. If f is **convex**, the only zero of derivative corresponds to the **unique minimizer** of f .

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}.$$

Minimizing a multivariate function with the Newton Method

Let $F : \mathbb{R}^n \mapsto \mathbb{R}$, $\nabla F = \left(\frac{\partial F}{\partial x_i} \right) : \mathbb{R}^n \mapsto \mathbb{R}^n$ be the **gradient** of F and $\mathcal{H} = \left(\frac{\partial^2 F}{\partial x_i^2} \right)$ be the **Hessian** of F .

$$x^{[k+1]} = x^{[k]} - \mathcal{H}(x^{[k]})^{-1} \nabla F(x^{[k]}).$$

Path following algorithm

A first (inefficient) idea

Fix μ_0

- Let $\mu_i = \alpha\mu_{i-1}$ with $0 < \alpha < 1$
- Compute $x(\mu_i)$ using the Newton method (with $x(\mu_{i-1})$ as a **starting point**)

Main difficulty : why solving a problem quite hard at each iteration ?

The path following idea

Fix μ_0

- Let $\mu_i = \alpha\mu_{i-1}$ with $0 < \alpha < 1$
- Compute an approximation of $x(\mu_i)$ using **one step** of the Newton method (with $x(\mu_{i-1})$ as a **starting point**)

Path following algorithm

A first (inefficient) idea

Fix μ_0

- Let $\mu_i = \alpha\mu_{i-1}$ with $0 < \alpha < 1$
- Compute $x(\mu_i)$ using the Newton method (with $x(\mu_{i-1})$ as a **starting point**)

Main difficulty : why solving a problem quite hard at each iteration ?

The path following idea

Fix μ_0

- Let $\mu_i = \alpha\mu_{i-1}$ with $0 < \alpha < 1$
- Compute an **approximation** of $x(\mu_i)$ using **one step** of the Newton method (with $x(\mu_{i-1})$ as a **starting point**)

The primal-dual path following algorithm

We will follow the primal and the dual central paths simultaneously.

The **target** $(x(\mu), p(\mu), s(\mu))$ (ignoring nonnegativity) at each iteration $k + 1$ satisfies

$$\begin{aligned}Ax &= b \\ A^T p + s &= c \\ x_i s_i &= \mu_{k+1} \quad \text{for all } i\end{aligned}$$

Stated otherwise we look for the zeros of the function

$$F(x, p, s) = (Ax - b, \quad A^T p + s - c, \quad XSe - \mu e)^T,$$

where $X = \text{diag}(x_1, \dots, x_n)$, $S = \text{diag}(s_1, \dots, s_m)$, $e = (1, \dots, 1)^T$.

Assume $(x^{[k]}, p^{[k]}, s^{[k]})$ is the **current iterate**. The **Jacobian** of F at $(x^{[k]}, p^{[k]}, s^{[k]})$ writes

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^{[k]} & 0 & X^{[k]} \end{pmatrix}$$

The current **residue** is

$$(0, \quad , 0, \quad X^{[k]} S^{[k]} - \mu_{k+1} e)$$

The primal-dual path following algorithm

We will follow the primal and the dual central paths simultaneously.

The **target** $(x(\mu), p(\mu), s(\mu))$ (ignoring nonnegativity) at each iteration $k + 1$ satisfies

$$\begin{aligned}Ax &= b \\ A^T p + s &= c \\ x_i s_i &= \mu_{k+1} \quad \text{for all } i\end{aligned}$$

Stated otherwise we look for the zeros of the function

$$F(x, p, s) = (Ax - b, \quad A^T p + s - c, \quad XSe - \mu e)^T,$$

where $X = \text{diag}(x_1, \dots, x_n)$, $S = \text{diag}(s_1, \dots, s_m)$, $e = (1, \dots, 1)^T$.

Assume $(x^{[k]}, p^{[k]}, s^{[k]})$ is the **current iterate**. The **Jacobian** of F at $(x^{[k]}, p^{[k]}, s^{[k]})$ writes

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^{[k]} & 0 & X^{[k]} \end{pmatrix}$$

The current **residue** is

$$(0, \quad , 0, \quad X^{[k]} S^{[k]} - \mu_{k+1} e)$$

The primal-dual path following algorithm

We will follow the primal and the dual central paths simultaneously.

The **target** $(x(\mu), p(\mu), s(\mu))$ (ignoring nonnegativity) at each iteration $k + 1$ satisfies

$$\begin{aligned}Ax &= b \\ A^T p + s &= c \\ x_i s_i &= \mu_{k+1} \quad \text{for all } i\end{aligned}$$

Stated otherwise we look for the zeros of the function

$$F(x, p, s) = (Ax - b, \quad A^T p + s - c, \quad XSe - \mu e)^T,$$

where $X = \text{diag}(x_1, \dots, x_n)$, $S = \text{diag}(s_1, \dots, s_m)$, $e = (1, \dots, 1)^T$.

Assume $(x^{[k]}, p^{[k]}, s^{[k]})$ is the **current iterate**. The **Jacobian** of F at $(x^{[k]}, p^{[k]}, s^{[k]})$ writes

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^{[k]} & 0 & X^{[k]} \end{pmatrix}$$

The current **residue** is

$$(0, \quad , 0, \quad X^{[k]} S^{[k]} - \mu_{k+1} e)$$

The primal-dual path following algorithm

We will follow the primal and the dual central paths simultaneously.

The **target** $(x(\mu), p(\mu), s(\mu))$ (ignoring nonnegativity) at each iteration $k + 1$ satisfies

$$\begin{aligned}Ax &= b \\ A^T p + s &= c \\ x_i s_i &= \mu_{k+1} \quad \text{for all } i\end{aligned}$$

Stated otherwise we look for the zeros of the function

$$F(x, p, s) = (Ax - b, \quad A^T p + s - c, \quad XSe - \mu e)^T,$$

where $X = \text{diag}(x_1, \dots, x_n)$, $S = \text{diag}(s_1, \dots, s_m)$, $e = (1, \dots, 1)^T$.

Assume $(x^{[k]}, p^{[k]}, s^{[k]})$ is the **current iterate**. The **Jacobian** of F at $(x^{[k]}, p^{[k]}, s^{[k]})$ writes

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^{[k]} & 0 & X^{[k]} \end{pmatrix}$$

The current **residue** is

$$(0, \quad , 0, \quad X^{[k]} S^{[k]} - \mu_{k+1} e)$$

Iterations of the primal-dual path following algorithm

To compute one step of the Newton Method to come closer to the target, we solve the **linear system**

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^{[k]} & 0 & X^{[k]} \end{pmatrix} \begin{pmatrix} \Delta x^{[k]} \\ \Delta p^{[k]} \\ \Delta s^{[k]} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mu_{k+1} e - X^{[k]} S^{[k]} \end{pmatrix} \quad (1)$$

The complete algorithm is therefore

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and $\mu_0 > 0$

For $i = 0, \dots$

- Set $\mu_{i+1} = \alpha \mu_i$
- Compute $(\Delta x^{[i]}, \Delta p^{[i]}, \Delta s^{[i]})$ by solving (1)
- Set $(x^{[i+1]}, p^{[i+1]}, s^{[i+1]}) = (x^{[i]}, p^{[i]}, s^{[i]}) + (\Delta x^{[i]}, \Delta p^{[i]}, \Delta s^{[i]})$

Iterations of the primal-dual path following algorithm

To compute one step of the Newton Method to come closer to the target, we solve the **linear system**

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S^{[k]} & 0 & X^{[k]} \end{pmatrix} \begin{pmatrix} \Delta x^{[k]} \\ \Delta p^{[k]} \\ \Delta s^{[k]} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \mu_{k+1} e - X^{[k]} S^{[k]} \end{pmatrix} \quad (1)$$

The complete algorithm is therefore

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and $\mu_0 > 0$

For $i = 0, \dots$

- Set $\mu_{i+1} = \alpha \mu_i$
- Compute $(\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$ by solving (1)
- Set $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility** ? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints** ?
- **Duality** ? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path ?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility** ? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints** ?
- **Duality** ? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path ?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility**? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints**?
- **Duality**? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility**? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints**?
- **Duality**? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility**? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints**?
- **Duality**? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility**? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints**?
- **Duality**? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

Analysis of the correctness of the primal-dual path following algorithm

- Are the iterates $(x^{[k]}, p^{[k]}, s^{[k]})$ sufficiently **close to the actual central path** i.e. close to $(x(\mu_k), p(\mu_k), s(\mu_k))$?
- A property of the point on the central path is $X^{[k]}S^{[k]}e = \mu_k e$
- We measure the **closeness to the central path** by

$$\delta(x, s, \mu) = \frac{1}{\mu} \|XSe - \mu e\|$$

Remark that if we ask for $\delta(x^{[k]}, s^{[k]}, \mu_k) < R$, the condition becomes **more restrictive** when μ decreases.

- If $\delta(x^{[k]}, s^{[k]}) < \tau$ do we have $\delta(x^{[k+1]}, s^{[k+1]}) < \tau$
- **Strict feasibility**? Can we prove that each iterate $(x^{[k]}, p^{[k]}, s^{[k]})$ satisfy the **positivity constraints**?
- **Duality**? Can we prove that the iterates have the right **duality measure** μ corresponding to the central path?

By choosing $\alpha = 1 - \frac{0.4}{\sqrt{n}}$ and $\tau = 0.4$, all these properties can be proven and if the **stopping criterion** is $n\mu_k < \epsilon$ then we need $\mathcal{O}(\sqrt{n} \log \frac{n\mu_0}{\epsilon})$ iterations until optimality is reached.

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close to 1**. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close** to 1. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close** to 1. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close** to 1. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k(\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close** to 1. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k(\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close** to 1. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k(\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

From short steps to long steps

- The previous algorithm has the best **theoretical performance** but is pretty slow in practice because α has to be chosen **very close** to 1. Therefore it is a **short step** method.
- Idea : try to reduce μ more aggressively in order to obtain longer steps. This is the **long step** method.
- Drawback : if the step of the Newton Method is too long, the new iterate **may not be feasible** (because it does not satisfy positivity).
- In order to restore **feasibility**, we have to **reduce the Newton step**

$$(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k(\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$$

with $\rho_k < 1$

- The new iterates do not satisfy the **right duality measure**.
- The new iterates may be **far from the central path**
- In this case, we apply several Newton steps with the same μ_k before decreasing further μ_k .

The primal-dual path following algorithm with long steps

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and μ_0

- Compute the Newton step $(\Delta x^{[k]}, \Delta p^{[k]}, s^{[k]})$
- Compute $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$
with ρ_k chosen such that $(x^{[k+1]}, p^{[k]}, s^{[k]}) > 0$
- If $\delta(x^{[k+1]}, s^{[k+1]}, \alpha \mu_k) < \tau$ set $\mu_{k+1} = \alpha \mu_k$ else set $\mu_{k+1} = \mu_k$

until $n\mu_{k+1} < \epsilon$.

By choosing α (for example = 0.5 or 0.1 or 0.01), an appropriate choice of τ and ρ_k allows the algorithm to terminate in $\mathcal{O}(n \log \frac{n\mu_0}{\epsilon})$ iterations.

Less efficient in theory

More efficient in practice

The primal-dual path following algorithm with long steps

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and μ_0

- Compute the Newton step $(\Delta x^{[k]}, \Delta p^{[k]}, s^{[k]})$
- Compute $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$
with ρ_k chosen such that $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) > 0$
- If $\delta(x^{[k+1]}, s^{[k+1]}, \alpha \mu_k) < \tau$ set $\mu_{k+1} = \alpha \mu_k$ else set $\mu_{k+1} = \mu_k$

until $n\mu_{k+1} < \epsilon$.

By choosing α (for example = 0.5 or 0.1 or 0.01), an appropriate choice of τ and ρ_k allows the algorithm to terminate in $\mathcal{O}(n \log \frac{n\mu_0}{\epsilon})$ iterations.

Less efficient in theory

More efficient in practice

The primal-dual path following algorithm with long steps

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and μ_0

- Compute the Newton step $(\Delta x^{[k]}, \Delta p^{[k]}, s^{[k]})$
- Compute $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$
with ρ_k chosen such that $(x^{[k+1]}, p^{[k]}, s^{[k]}) > 0$
- If $\delta(x^{[k+1]}, s^{[k+1]}, \alpha \mu_k) < \tau$ set $\mu_{k+1} = \alpha \mu_k$ else set $\mu_{k+1} = \mu_k$

until $n\mu_{k+1} < \epsilon$.

By choosing α (for example = 0.5 or 0.1 or 0.01), an appropriate choice of τ and ρ_k allows the algorithm to terminate in $\mathcal{O}(n \log \frac{n\mu_0}{\epsilon})$ iterations.

Less efficient in theory

More efficient in practice

The primal-dual path following algorithm with long steps

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and μ_0

- Compute the Newton step $(\Delta x^{[k]}, \Delta p^{[k]}, s^{[k]})$
- Compute $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$
with ρ_k chosen such that $(x^{[k+1]}, p^{[k]}, s^{[k]}) > 0$
- If $\delta(x^{[k+1]}, s^{[k+1]}, \alpha \mu_k) < \tau$ set $\mu_{k+1} = \alpha \mu_k$ else set $\mu_{k+1} = \mu_k$

until $n\mu_{k+1} < \epsilon$.

By choosing α (for example = 0.5 or 0.1 or 0.01), an appropriate choice of τ and ρ_k allows the algorithm to terminate in $\mathcal{O}(n \log \frac{n\mu_0}{\epsilon})$ iterations.

Less efficient in theory

More efficient in practice

The primal-dual path following algorithm with long steps

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and μ_0

- Compute the Newton step $(\Delta x^{[k]}, \Delta p^{[k]}, s^{[k]})$
- Compute $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$
with ρ_k chosen such that $(x^{[k+1]}, p^{[k]}, s^{[k]}) > 0$
- If $\delta(x^{[k+1]}, s^{[k+1]}, \alpha\mu_k) < \tau$ set $\mu_{k+1} = \alpha\mu_k$ else set $\mu_{k+1} = \mu_k$

until $n\mu_{k+1} < \epsilon$.

By choosing α (for example = 0.5 or 0.1 or 0.01), an appropriate choice of τ and ρ_k allows the algorithm to terminate in $\mathcal{O}(n \log \frac{n\mu_0}{\epsilon})$ iterations.

Less efficient in theory

More efficient in practice

The primal-dual path following algorithm with long steps

Start with $(x^{[0]}, p^{[0]}, s^{[0]})$ and μ_0

- Compute the Newton step $(\Delta x^{[k]}, \Delta p^{[k]}, s^{[k]})$
- Compute $(x^{[k+1]}, p^{[k+1]}, s^{[k+1]}) = (x^{[k]}, p^{[k]}, s^{[k]}) + \rho_k (\Delta x^{[k]}, \Delta p^{[k]}, \Delta s^{[k]})$
with ρ_k chosen such that $(x^{[k+1]}, p^{[k]}, s^{[k]}) > 0$
- If $\delta(x^{[k+1]}, s^{[k+1]}, \alpha\mu_k) < \tau$ set $\mu_{k+1} = \alpha\mu_k$ else set $\mu_{k+1} = \mu_k$

until $n\mu_{k+1} < \epsilon$.

By choosing α (for example = 0.5 or 0.1 or 0.01), an appropriate choice of τ and ρ_k allows the algorithm to terminate in $\mathcal{O}(n \log \frac{n\mu_0}{\epsilon})$ iterations.

Less efficient in theory

More efficient in practice

Extension to conic programming

Interior point methods are the classic way to solve **Conic quadratic** and **semidefinite** programming.

To be able to extend it, we need to define a **barrier** and to write a characterization of the **central path**

- Barrier for the conic quadratic case

$$-\log(x_0^2 - x_1^2 - \dots - x_n^2)$$

- Barrier for the semidefinite case

$$-\log(\det(X))$$

- Characterization of the central path in the semidefinite case
 - primal feasibility
 - dual feasibility
 - modified **complementarity slackness** $X^{[k]}S^{[k]} = \mu_k$

Extension to conic programming

Interior point methods are the classic way to solve **Conic quadratic** and **semidefinite** programming.

To be able to extend it, we need to define a **barrier** and to write a characterization of the **central path**

- Barrier for the conic quadratic case

$$-\log(x_0^2 - x_1^2 - \dots - x_n^2)$$

- Barrier for the semidefinite case

$$-\log(\det(X))$$

- Characterization of the central path in the semidefinite case
 - primal feasibility
 - dual feasibility
 - modified **complementarity slackness** $X^{[k]}S^{[k]} = \mu_k$

Extension to conic programming

Interior point methods are the classic way to solve **Conic quadratic** and **semidefinite** programming.

To be able to extend it, we need to define a **barrier** and to write a characterization of the **central path**

- Barrier for the conic quadratic case

$$-\log(x_0^2 - x_1^2 - \dots - x_n^2)$$

- Barrier for the semidefinite case

$$-\log(\det(X))$$

- Characterization of the central path in the semidefinite case
 - primal feasibility
 - dual feasibility
 - modified **complementarity slackness** $X^{[k]}S^{[k]} = \mu_k$