- Equality constrained minimization:

\[
\begin{align*}
\min_{x} & \quad f(x) \\
\text{s.t.} & \quad Ax = b \\
\Rightarrow & \quad Ax^* = b \quad \text{and} \quad \nabla f(x^*) + A^T \nu^* = 0
\end{align*}
\]

- Special case:

\[
\begin{align*}
\min_{x} & \quad \frac{1}{2} x^T P x + q^T x + \frac{1}{2} \\
\text{s.t.} & \quad Ax = b \\
\Rightarrow & \quad \begin{bmatrix} P & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \nu^* \end{bmatrix} = \begin{bmatrix} -q \\ b \end{bmatrix} \\
& \quad \text{(KKT)}
\end{align*}
\]

- Elimination Technique:

\[
\begin{align*}
\{ x \mid Ax = b \} &= \{ x \mid x = Fz + \hat{x} \text{ for some } z \in \mathbb{R}^n \}
\end{align*}
\]

- Newton's Method with equality constraints:

**Question:** Given \( x^{(k)} \) satisfying \( Ax = b \), how to find \( \Delta x \)?

**Newton Step**

- KKT:

\[
\begin{align*}
A(x^{(k)} + \Delta x) &= b \\
\nabla f(x^{(k)}) + A^T \nu &= 0
\end{align*}
\]

\[
\Rightarrow \quad \begin{bmatrix} \nabla^2 f(x^{(k)}) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \nu \end{bmatrix} = \begin{bmatrix} -\nabla f(x^{(k)}) \\ 0 \end{bmatrix}
\]
Note: If \( x^{(k)} \) is feasible and \( A\Delta x = 0 \), then \( x^{(k+1)} = x^{(k)} + \alpha \Delta x \) is feasible for all \( \alpha \)'s. \( \Rightarrow \) At each iteration, we find a feasible point. \( \Rightarrow \) Feasible descent direction.

Infeasible start Newton method:
\( x^{(k)} \) may not satisfy \( Ax^{(k)} = b \).

\[
\begin{bmatrix}
\nabla^2 f(x^{(k)}) & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta v
\end{bmatrix} = -
\begin{bmatrix}
\nabla f(x^{(k)}) \\
Ax^{(k)} - b
\end{bmatrix}
\]

\( \Rightarrow \) \( (x^{(k)} + \Delta x) \) is feasible but \( x^{(k)} + t \Delta x \) may be infeasible.

(previous idea: choose \( t \) such that \( f(x^{(k+1)}) < f(x^{(k)}) \))

Primal-dual Newton step: iterate on both primal and dual parameters: \( (x^{(k)} , v^{(k)}) \rightarrow (x^{(k+1)} , v^{(k+1)}) \)

\[
A(x^{(k)} + \Delta x) = b , \quad \nabla f(x^{(k)} + \Delta x) + A^T(v^{(k)} + \Delta v) = 0
\]

\[
\Rightarrow \begin{bmatrix}
\nabla^2 f(x^{(k)}) & A^T \\
A & 0
\end{bmatrix}
\begin{bmatrix}
\Delta x \\
\Delta v
\end{bmatrix} = -
\begin{bmatrix}
\nabla f(x^{(k)}) + A^T v^{(k)} \\
Ax^{(k)} - b
\end{bmatrix}
\]

Measure of optimality: \( y(x,v) = [Ax-b, \nabla f(x) + A^T v] \) residual

Step size: choose \( t \) such that \( \| y(x^{(k)} + t \Delta x, v^{(k)} + t \Delta v) \| \) is minimum.
- Step size is based on residual, instead of \( f(x^n) \).

- Interior point method:

\[
\begin{align*}
\min \ f_0(x) \\
\text{s.t.} \ f_i(x) \leq 0 \\
Ax = b
\end{align*}
\]

\[
\begin{align*}
\lambda^* > 0 \\
\nabla f_0(x^*) &\leq \sum \lambda_i^* \nabla f_i(x^*) + A^T \lambda^* = 0 \\
\lambda_i^* f_i(x^*) &= 0
\end{align*}
\]

\[
\min \ f_0(x) + \sum \lambda_i^* f_i(x) \\
\text{s.t.} \ Ax = b
\]

\[
\begin{align*}
I_u = \frac{1}{t} \log (-t) \\
\log \text{approximation}
\end{align*}
\]

- Quality of approximation improves as \( t \) grows.

- Define: \( \phi(x) = \sum \log (-f_i(x)) \)

\[
\begin{align*}
\min \ x f_0(x) + \phi(x) \\
\text{s.t.} \ Ax = b
\end{align*}
\]

- Thm: \( x^*(t) \) satisfies modified LICIT where \( \lambda_i f_i(x) \) is replaced by \( \lambda_i f_i(x) = \frac{1}{t} \).
Two observations:

1. For every $t$, $x^*(t)$ is feasible for original opt.
2. As $t \to \infty$, $x^*(t)$ goes to $x^*$.

$\Rightarrow$ we have central path:

\[ f_0(x^*(t)) - p^* \leq \frac{m}{t} \]

(m: # of constraints)

(because in the dual: $\sum_{i=1}^{m} \lambda_i^* f_i(x^*_i(t)) = \frac{m}{t}$)

$\Rightarrow$ Given $t$, we know the quality of approximation.

Algorithm: Given strictly feasible $x$, $t^{(0)} > 0$ and tolerance $\varepsilon > 0$.

- Repeat:
  - Centering step:
    - Compute $x^*(t)$ by $\min \{ t \text{ for } \emptyset \} \text{ by starting at } x^*$.
    - $Ax = b$
  - Update: $x = x^*(t)$
  - Stopping criterion: quit if $\frac{m}{t} < \varepsilon$
  - Increase $t$: $t = t \mu$ (for some $\mu > 1$)

Comment: Finding $x^*(t)$ accurately is not important.
- Finding $x^*(t)$ needs a few iterations given its initialization.

Question: How to find a strictly feasible $x$ to start with?
Phase I: Solve an optimization to find \( x \):

\[
\min \quad \frac{\lambda}{s} \\
\text{s.t. } f(x) < s \\
Ax = b
\]

- Choose \( x \) and a big \( s \) to make it feasible.
- Use barrier method to find an \( x \) corresponding to \( s < 0 \).

Summary: We have inner and outer loop.
- Inner loop finds a point on the central path.
- Outer loop updates \( t \).

Question: Can we merge these inner and outer iterations?

Modified KKT:

\[
\nabla f_0(x) + \sum_{i} (f(x_i))^T \lambda + A^T \nu = 0 \\
\text{diag}(\lambda) f(x) - \frac{1}{t} = 0 \\
A x - b = 0
\]

\( \lambda, I(f(x)) \) = Jacobian

Define \( y = \begin{bmatrix} x^T \lambda^T \end{bmatrix} \). \( \Rightarrow \) Iterate on \( y \), i.e., on both primal and dual parameters.

Direction update: find \( \Delta y \) such that \( y + \Delta y \) satisfies linearized modified KKT.

Step size update: find \( \alpha \) such that \( y + \alpha \Delta y \) gives the smallest residual for the left side of KKT.

Determine \( t \): increase it by a factor of \( \mu > 1 \).

Note: Step size should be close to guarantee \( \lambda > 0 \), \( f(x) < 0 \).
Distributed computation

- \[ \min_{x_1, x_2} f_1(x_1) + f_2(x_2) \Rightarrow \nabla f_1(x_1) = 0 \text{ and } \nabla f_2(x_2) = 0 \]
  "no coupling"

- \[ \min_{x_1, x_2, y} f_1(x_1, y) + f_2(x_2, y) \Rightarrow \nabla_x f_1(x_1, y) = 0, \nabla_{x_2} f_2(x_2, y) = 0 \]
  \[ \nabla_y f_1(x_1, y) + \nabla_y f_2(x_2, y) = 0 \]

- Consider three entities for \( x_1, x_2 \) and \( y \).

\[ \begin{array}{ccc}
1 & \xrightarrow{\text{min}} & 2 \\
\xrightarrow{\text{solve}} & y & \xrightarrow{\text{return}} x_i^{(k)} \\
\xrightarrow{\text{update}} & y^{(k+1)} & = y^{(k)} - t \times \left( \nabla_y f_1(x_1^{(k)}, y^{(k)}) + \nabla_y f_2(x_2^{(k)}, y^{(k)}) \right) \\
\end{array} \]

This is called primal decomposition.

\( \Rightarrow \) The problem can be solved on multiple machines.

Different entities may not know private information of each other (say 1 doesn't need to know \( f_2(x_2, y) \)).

- Another approach:

\[ \min_{x_1, x_2, y} f_1(x_1, y) + f_2(x_2, y) \iff \min_{x_1, x_2, y} f_1(x_1, y) + f_2(x_2, y) \]

\[ y_1 = y_2 \]

Consistency constraint.
→ let $\lambda$ be the dual param for $\gamma_1=\gamma_2$.

⇒ Algorithm:
- 1 solves $\min_{x_i, y_i} f_i(x_i, y_i) + \lambda^{(k)} y_i$
- 2 solves $\min_{x_2, y_2} f_2(x_2, y_2) - \lambda^{(k)} y_2$
- 3 returns $(x_i^{(k)}, y_i^{(k)})$ to 3 for $i=1, 2$
- 3 updates $\lambda^{(k+1)}$ as $\lambda^{(k)} + \varepsilon_i (y_i - \gamma_i)$

- More complicated example:
  \[
  \min_{x_1, x_2} f_1(x_1) + f_2(x_2) \quad \text{subject to} \quad h_1(x_1) + h_2(x_2) \leq 0
  \]

⇒ 3 propagates $\lambda$
- 1 solves $\min_{x_i} f_i(x_i) + \lambda_i h_i(x_i)$ for $i=1, 2$
- 3 updates $\lambda$ based on $h_1(x_1) + h_2(x_2)$ at the new $x_1$ and $x_2$.

- General decomposition structure:
  - multiple subsystems
  - coupling constraints between subsystems
  - we represent the structure by a graph

\[ \begin{array}{c}
S_1 \quad y \quad S_2 \quad z \quad S_3 \\
\text{x_1} \quad \text{public variable} \quad \text{x_2} \quad \text{private variable} \quad \text{x_3}
\end{array} \]

\[ \begin{align*}
\min f_1(x_1, y) + f_2(x_1, y, z) + f_3(x_3, z) \\
\text{s.t.} \quad (x_1, y) \in C_1, \quad (x_1, y, z) \in C_2, \quad (x_3, z) \in C_3
\end{align*} \]

⇒ Replace $y$ by $y_1$ and $y_2$ s.t. $y_1 = y_2$ Replace $z$ by $z_2$ and $z_3$ s.t. $z_2 = z_3$
Congestion control for Comm networks:

- There is a network of links with fixed capacities.
- Each user wants to transmit data from a source to a destination over a fixed route.

$\lambda$: rate of user $i$, $y_L$: capacity of Link $L$

$U_i(\lambda)$: utility of user $i$.

- Problem of interest: $\max_{\lambda \in \mathbb{R}^+} \sum_{i \in \mathcal{S}} U_i(\lambda)$

( $\mathcal{S}$: set of users, $\mathcal{E}$: set of links, $\mathcal{S}(L)$: users using Link $L$)

- This convex problem can be solved easily.

- Challenge: There is no coordinator to collect the information and solve a centralized problem.

- Solution: Propose a distributed algorithm.

- Define: $y_L = \sum_{i \in \mathcal{S}(L)} \lambda_i$, $P_L$: dual variable for $y_L \leq C_L$

$\mu_i = \sum_{L \in \mathcal{E}(i)} P_L \rightarrow$ sum of dual params over a route

$\Rightarrow$ KKT:

\[ \begin{cases} 
U'(\lambda_i) = \mu_i \\
\sum_{L \in \mathcal{E}(i)} P_L (y_L - C_L) = 0 \\
y_L - C_L \leq 0 \\
\lambda_i \geq 0 \\
P_L \geq 0 
\end{cases} \]
Strategy: Each user updates $x_r$ based on $q_r$.
- Each link updates $P_L$ based on $y_L$.

$P_L$: price for using link $L$

$\Rightarrow$ $x_r$ is updated based on a route price $= P_L$

$y_L$ is updated based on the incoming rate $= x_r$

- Dual Algorithm:

$$
\begin{cases}
    - x_r(t) = (U_r')^{-1} (q_r(t)) \\
    - P_L(t) = \alpha (y_L(t) - c_L)^+ P_L(t)
\end{cases}
$$

$$(U)^+ = \begin{cases}
    U & \text{if } P > 0 \\
    \max(U, 0) & \text{if } P = 0
\end{cases}$$

Interpretation: If $y_L(t) > c_L \Rightarrow P_L(t)$ increases

$\Rightarrow q_L(t)$ increases

$\Rightarrow x_r(t)$ decreases $\Rightarrow y_L(t)$ decreases

If $y_L(t) < c_L \Rightarrow x_r(t)$ tries to increase.

$\Rightarrow$ With global coordination, each user and link make decisions based on local information.

- Primal Algorithm:

$$
\begin{cases}
    - \dot{x}_r(t) = \alpha (U_r (x_r(t)) - q_r(t)) \\
    - P_L(t) = f_c (y_L(t))
\end{cases}
$$

This is a fluid model as updates happen in the continuous domain $t$ (time).