Lecture 12: Descent Methods for Constrained Minimization, Manifold Sub-optimization Methods, Problems with Simple Constraints

Prof. Krishna R. Pattipati
Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna@engr.uconn.edu (860) 486-2890

ECE 6437
Computational Methods for Optimization

Fall 2009
November 17, 2009

Outline of Lecture 12

- Review: Multiplier (primal-dual and augmented Lagrangian methods) and Successive Quadratic Programming methods
- Feasible Direction Methods
  - Rosen's Gradient Projection methods
  - Reduced Gradient method
  - Newton-type methods
  - Updating QR decompositions
- Problems with Simple Constraints
- Subgradient Methods for Discrete Optimization
- Cutting plane methods

Review of Multiplier and SQP Methods

- Let us review the methods that we have discussed so far

Multiplier (primal-dual, AL) methods

$$\min f(\underline{x})$$
$$\text{s.t } \underline{h}(\underline{x}) = \underline{0}$$
$$\underline{g}(\underline{x}) \le \underline{0}$$

Form augmented Lagrangian

$$L_c(\underline{x},\underline{\lambda},\underline{\mu}) = f(\underline{x}) + \underline{\lambda}^T \underline{h}(\underline{x}) + \underline{\mu}^T \underline{g}^+(\underline{x})$$
$$\frac{c}{2}\underline{h}^T(\underline{x})\underline{h}(\underline{x}) + \frac{c}{2}\underline{g}^{+T}(\underline{x})\underline{g}^+(\underline{x})$$

$$g_j^+(\underline{x},\underline{\mu}) = \max\left(g_j(\underline{x}), {}^{-\mu_j}\!/_{c}\right)$$

Minimize $L_c(\underline{x},\underline{\lambda}_k,\underline{\mu}_k)$ w.r.t $\underline{x}$ to obtain $\underline{x}_k$

Update $\underline{\lambda}_k, \underline{\mu}_k$

- Key: Solve an unconstrained problem at each step

Successive Quadratic programming methods

- At each $\underline{x}_k, \underline{\lambda}_k, \underline{\mu}_k$ approximate Lagrangian function by a quadratic and constraints by a linear function
- Find direction $\underline{d}_k \ni$ it minimizes

$$\underline{d}^T B \underline{d} + \nabla \underline{f}^T(\underline{x}_k)\underline{d}$$
$$\text{s.t. } \nabla \underline{h}^T(\underline{x}_k)\underline{d} + \underline{h}(\underline{x}_k) = \underline{0}$$
$$\nabla \underline{g}^T(\underline{x}_k)\underline{d} + \underline{g}(\underline{x}_k) \le \underline{0}$$
$$\Rightarrow \underline{d}_k, \underline{\lambda}_{k+1}, \underline{\mu}_{k+1}$$

- Find stepsize α_k to min $f + cP$ along $\underline{d}_k$

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$$

- Key: Solve a special constrained problem at each step

Feasible Direction Methods

- Feasible Direction Methods

Manifold sub-optimization methods

Gradient projection

Reduced gradient

Quadratic programming (Newton-type Methods)

Manifold Suboptimization Methods - 1

- These methods are good for linear equality and inequality constraints:

$$\min f(\underline{x})$$

$$\text{s.t.}\quad A_1\underline{x} = \underline{b}_1 \qquad \underline{b}_1 \in R^m \Rightarrow \underline{a}_i^T\underline{x} = b_i,\ i = 1,2,..,m$$

$$A_2\underline{x} \le \underline{b}_2 \qquad \underline{b}_2 \in R^r \quad \Rightarrow \underline{a}_j^T\underline{x} \le b_j,\ i = 1,2,..,r$$

Suppose have a feasible $\underline{x}_k$. For example, such a point at k=0 can be obtained via the following *phase I of LP*

$$\min \sum_{i=1}^{m} y_i + \sum_{j=1}^{r} z_j$$

$$\text{s.t.}\quad A_1\underline{x} + \underline{y} = \underline{b}_1$$

$$A_2\underline{x} + \underline{z} = \underline{b}_2$$

$$\text{and}\quad \underline{y} \ge 0$$

$$\underline{z} \ge 0$$

$$\Rightarrow$$

Optimal solution is such that

$$\underline{y} = 0$$

$$\underline{z} \ge 0$$

Initial FS: $\underline{y} = 0$, $\underline{z} = \underline{b}_2$

assume WLOG $\underline{b}_2 \ge \underline{0}$

Manifold Suboptimization Methods - 2

$\Rightarrow$ In the process, we get $A_1\underline{x} = \underline{b}_1$

$$\hat{A}_2\underline{x} = \underline{\hat{b}}_2 \text{ for a subset of inequality constraints}$$

Let $\mathcal{A}(\underline{x}) = \left\{\underline{x} \mid \underline{a}_j^T\underline{x} = b_j,\ j = 1, 2, \ldots, r\right\}$ *Active set*

Suppose we are given a feasible $\underline{x}_k \ni A_1\underline{x}_k = \underline{b}_1$ & $\hat{A}_2\underline{x}_k = \underline{b}_2$ for some active inequality constraints. What are the feasible directions, $\underline{d}_k$?

If the new point is $\underline{x}_{k+1}$, then

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k\underline{d}_k$$

Know $\underbrace{\begin{pmatrix} A_1 \\ \hat{A}_2 \end{pmatrix}}_{A_k}\underline{x}_k = \underbrace{\begin{pmatrix} \underline{b}_1 \\ \underline{\hat{b}}_2 \end{pmatrix}}_{\underline{b}_k}$

$$A_k\underline{x}_k = \underline{b}_k$$

$$A_k\underline{x}_{k+1} = A_k\underline{x}_k + \alpha_k A_k\underline{d}_k$$

Manifold Suboptimization Methods - 3

Since $\underline{x}_{k+1}$ is feasible

$$\begin{aligned} A_1\underline{x}_{k+1} &= \underline{b}_1 \\ \hat{A}_2\underline{x}_{k+1} &\le \underline{\hat{b}}_2 \end{aligned} \quad \Rightarrow \quad \begin{aligned} A_1\underline{d}_k &= \underline{0} \\ \hat{A}_2\underline{d}_k &\le \underline{0} \end{aligned}$$

So, feasible directions must satisfy

$$\begin{pmatrix} A_1 \\ \hat{A}_2 \end{pmatrix} \underline{d}_k \quad \begin{pmatrix} = \underline{0} \\ \le \underline{0} \end{pmatrix}$$

In manifold suboptimization methods, we pick $\underline{d}_k$ such that

$$A_k\underline{d}_k = \underline{0}, \quad \mathcal{A}(\underline{x}_k) \sim \text{ active set on working set}$$

- Basic idea: Given a feasible point $\underline{x}_k$ and working set $\mathcal{A}(\underline{x}_k)$

$$\text{find } \underline{d}_k \ni A_k\underline{d}_k = \underline{0} \text{ and } \underline{g}_k^T\underline{d}_k < 0$$

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k\underline{d}_k$$

$$\text{Update } \mathcal{A}(\underline{x}_k) \to \mathcal{A}(\underline{x}_{k+1}) \Rightarrow A_{k+1}$$

Directions via QPP

- Key question:
  1. How to find $\underline{d}_k$? The problem is more general than just $\underline{g}_k^T \underline{d}_k < 0$
  2. How to update working sets?

  Most of the methods based on this idea can be viewed as one of solving a quadratic programming problem (QPP) of the form:

$$\min_{\underline{d}} \underline{g}_k^T \underline{d} + \frac{1}{2} \underline{d}^T M_k \underline{d}_k$$

$$\text{s.t.} \quad A_k \underline{d} = \underline{0}$$

Choice of M_k determines the type of projection algorithm

$$L(\underline{d}, \underline{\mu}) = \underline{g}_k^T \underline{d} + \frac{1}{2} \underline{d}^T M_k \underline{d}_k + \underline{\mu}^T A_k \underline{d}$$

$$\underline{d}_k = -M_k^{-1} \left(\underline{g}_k + A_k^T \underline{\mu} \right)$$

$$\underline{\mu} = -\left(A_k M_k^{-1} A_k^T \right)^{-1} A_k M_k^{-1} \underline{g}_k$$

$$\underline{d}_k = -M_k^{-1} \left[I - A_k^T \left(A_k M_k^{-1} A_k^T \right)^{-1} A_k M_k^{-1} \right] \underline{g}_k$$

M_k is a PD matrix on the subspace $\{\underline{d} : A_k \underline{d} = \underline{0}\} = N(A_k)$

$\Rightarrow \begin{bmatrix} M_K & A_k^T \\ A_k & 0 \end{bmatrix}$ is nonsingular $\Rightarrow$ unique $\underline{d}_k$

Use QR decomposition for numerical stability

Updating Working Set - 1

- $$\underline{d}_k \neq 0 \Rightarrow \underline{g}_k^T \underline{d}_k + \frac{1}{2}\underline{d}_k^T M_k \underline{d}_k \leq 0$$

$$\Rightarrow \underline{g}_k^T \underline{d}_k \leq -\frac{1}{2}\underline{d}^T M_k \underline{d}_k < 0 \Rightarrow \text{ feasible}$$

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$$

$$\alpha_k = \begin{cases} \min\limits_i \dfrac{b_i - \underline{a}_i^T \underline{x}_k}{\underline{a}_i^T \underline{d}_k}, & i \notin \mathcal{A}(\underline{x}_k) \;\&\; \underline{a}_i^T \underline{d}_k > 0 \\ \infty, & \text{if for all } i \notin \mathcal{A}(\underline{x}_k),\; \underline{a}_i^T \underline{d}_k < 0 \end{cases}$$

Then, add constraint i_a to $\mathcal{A}(\underline{x}_k)$

$$\mathcal{A}(\underline{x}_{k+1}) = \mathcal{A}(\underline{x}_k) \cup \{i_a\}$$

$$i_a = \arg\min_i \frac{b_i - \underline{a}_i^T \underline{x}_k}{\underline{a}_i^T \underline{d}_k} \ni \underline{a}_i^T \underline{d}_k > 0 \;\& i \notin \mathcal{A}(\underline{x}_k)$$

Updating Working Set - 2

- $\underline{d}_k = \underline{0} \Rightarrow$ cannot move

$$A_k^T \underline{\mu} = -\underline{g}_k$$

$$\underline{g}_k + \underbrace{\sum_{i=1}^{m} \lambda_i \underline{a}_{1i}}_{\mu^s \text{ of equality}} + \sum_{j \in \mathcal{A}(\underline{x}_k)} \mu_j \underline{a}_{2j} = \underline{0}$$

if $\mu_j > 0$ for all $j \notin \mathcal{A}(\underline{x}_k) \Rightarrow$ optimal

Otherwise, drop the constraint with the most negative μ_j.

$$\mathcal{A}(\underline{x}_{k+1}) = \mathcal{A}(\underline{x}_k) - \{i_d\}$$

$$i_d = \arg\min_{j:\mu_j<0} \{\mu_j\}$$

QR Implementation

$$\text{Let } A_k^T = \begin{pmatrix} Q_k & \bar{Q}_k \end{pmatrix} \begin{pmatrix} R_k \\ 0 \end{pmatrix} = Q_k R_k$$

$$\underset{\textit{spans } R(A_k^T)}{Q_k \uparrow} \qquad \underset{\textit{spans } N(A_k)}{\bar{Q}_k \uparrow}$$

To find $\underline{d}_k$, need to solve

$$\begin{pmatrix} M_k & A_k^T \\ A_k & 0 \end{pmatrix} \begin{pmatrix} \underline{d}_k \\ \underline{\mu}_{k+1} \end{pmatrix} = \begin{pmatrix} -\underline{g}_k \\ \underline{0} \end{pmatrix} \quad \Rightarrow A_k^T = Q_k R_k$$

$$\text{Let } \underline{d}_k = Q_k \underline{c}_k + \bar{Q}_k \underline{a}_k \Rightarrow \begin{pmatrix} M_k Q_k & M_k \bar{Q}_k & Q_k R_k \\ R_k^T & 0 & 0 \end{pmatrix} \begin{pmatrix} \underline{c}_k \\ \underline{a}_k \\ \underline{\mu}_{k+1} \end{pmatrix} = \begin{pmatrix} -\underline{g}_k \\ \underline{0} \end{pmatrix}$$

$$\underline{c}_k = Q_k^T \underline{d}_k \ \& \ R_k^T \underline{c}_k = \underline{0} \Rightarrow \underline{c}_k = \underline{0} \Rightarrow \underline{d}_k = \bar{Q}_k \underline{a}_k$$

$$\bar{Q}_k^T M_k \bar{Q}_k \underline{a}_k = -\bar{Q}_k^T \underline{g}_k$$

$$\text{Do Cholesky on } \bar{Q}_k^T M_k \bar{Q}_k = \bar{U}_k \bar{U}_k^T \Rightarrow \underline{a}_k = -\left(\bar{U}_k \bar{U}_k^T\right)^{-1} \bar{Q}_k^T \underline{g}_k \quad ;$$

$$\underline{d}_k = -\bar{Q}_k \left(\bar{U}_k \bar{U}_k^T\right)^{-1} \bar{Q}_k^T \underline{g}_k \quad ; \quad \underline{g}_k^T \underline{d}_k = -\underline{g}_k^T \bar{Q}_k \left(\bar{U}_k \bar{U}_k^T\right)^{-1} \bar{Q}_k^T \underline{g}_k < 0$$

$\boxed{\text{Descent Direction}}$

Rosen's Gradient Projection Algorithm

- Choice of M_k determines the type of projection algorithm
  - Rosen's Gradient Projection Algorithm $M_k = I$

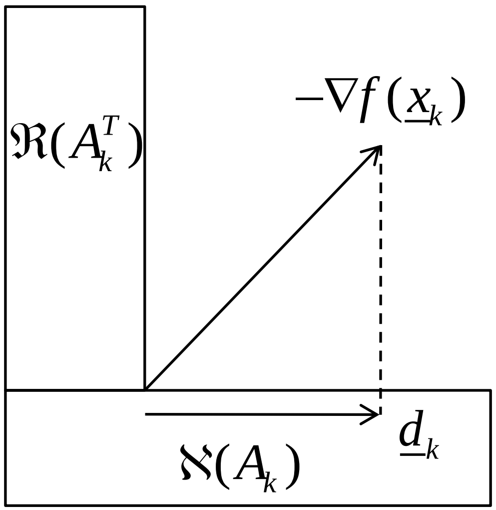
$$\underline{\mu} = -\left(A_k A_k^T\right)^{-1} A_k \underline{g}_k$$

complementary projection

$$\underline{d}_k = -\left[I - A_k^T\left(A_k A_k^T\right)^{-1} A_k\right]\underline{g}_k = -P_k^c \underline{g}_k$$

$$\Rightarrow \min \frac{1}{2} \| \underline{d} + \underline{g}_k \|^2 \ s.t.\ A_k \underline{d} = \underline{0}$$

$\underline{d}_k$ = Projection of $-\underline{g}_k$ onto the null space of A_k corresponding to active constraints



so,

$$\underline{x}_{k+1} = \underline{x}_k - P_k^c \underline{g}_k$$

In terms of QR decompistion,

$$\underline{d}_k = -\bar{Q}_k\left[\bar{Q}_k^T \bar{Q}_k\right]^{-1} \bar{Q}_k^T \underline{g}_k = -P_k^c \underline{g}_k$$

$$Example: A_k = \underline{e}^T$$

$$\underline{d}_k = (I - \frac{\underline{e}\underline{e}^T}{n})\underline{g}_k = \underline{g}_k - \underline{e} g_{av}, \quad g_{av} = \frac{\sum_{i=1}^{n} g_{ki}}{n}$$

Reduced Gradient Method

- Reduced Gradient Method: Select $M_k = \bar{Q}_k \left(\bar{Q}_k^T \bar{Q}_k\right)^{-2} \bar{Q}_k^T$

$$\Rightarrow \bar{Q}_k^T M_k \bar{Q}_k = I$$

$$\Rightarrow d_k = -\bar{Q}_k \bar{Q}_k^T \underline{g}_k$$

The update equation is:

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k \bar{Q}_k \bar{Q}_k^T \underline{g}_k$$

- Reduced gradient method can be viewed as a steepest descent iteration for solving the problem of minimizing $f(\underline{x})$ over the manifold $\{\underline{x} : A_k(\underline{x} - \underline{x}_k) = \underline{0}\}$ i.e., $\underline{x} = \underline{x}_k + Q_k \underline{a}$

$$\left.\begin{aligned} &\min \; f(\underline{x}) \\ &\text{s.t.} \;\; A_k(\underline{x} - \underline{x}_k) = \underline{0} \end{aligned}\right\} \Rightarrow \begin{aligned} &\min \; h(\underline{a}) = f\left(\underline{x}_k + \bar{Q}_k \underline{a}\right) \\ &\text{s.t.} \;\; \underline{a} \in R^{n-m-\hat{r}} \end{aligned}$$

$$\Rightarrow \underline{a}_{k+1} = \underline{a}_k - \alpha_k \nabla \underline{h}\left(\underline{a}_k\right)$$

$$= -\alpha_k \bar{Q}_k^T \nabla \underline{f}\left(\underline{x}_k\right) \quad \text{when } \underline{a}_k = \underline{0}$$

$$\text{when } \underline{a}_k = 0 \Rightarrow \underline{a}_{k+1} = -\alpha_k \bar{Q}_k^T \nabla \underline{f}\left(\underline{x}_k\right) = -\alpha_k \bar{Q}_k^T \underline{g}_k$$

$$\therefore \boxed{\underline{x}_{k+1} = \underline{x}_k + \bar{Q}_k \underline{a}_{k+1} = \underline{x}_k - \alpha_k \bar{Q}_k \bar{Q}_k^T \underline{g}_k}$$

Newton-type Methods

- Newton-type Methods:

$$\underline{a}_{k+1} = \underline{a}_k - \alpha_k D_k \nabla \underline{h}(\underline{a}_k) = \underline{a}_k - \alpha_k D_k \bar{Q}_k^T \underline{g}_k$$

$$D_k \sim \text{approximation to } \left[\nabla^2 h(\underline{a}_k)\right]^{-1}$$

when $\underline{a}_k = \underline{0}$

$$\underline{a}_{k+1} = -\alpha_k D_k \bar{Q}_k^T \underline{g}_k$$

and $\underline{x}_{k+1} = \underline{x}_k + \bar{Q}_k \underline{a}_{k+1}$

$$= \underline{x}_k - \alpha_k \bar{Q}_k D_k \bar{Q}_k^T \underline{g}_k$$

so, $\quad d_k = -\bar{Q}_k D_k \bar{Q}_k^T \underline{g}_k$

Note that this is similar to

$$D_k = \left(\bar{Q}_k^T M_k \bar{Q}_k\right)^{-1} \Rightarrow M_k = \bar{Q}_k \left(\bar{Q}_k^T \bar{Q}_k\right)^{-1} D_k^{-1} \left(\bar{Q}_k^T \bar{Q}_k\right)^{-1} \bar{Q}_k^T$$

when $D_k = \left[\nabla^2 h(\underline{a}_k)\right]^{-1} = \left[\bar{Q}_k^T \nabla^2 f(\underline{x}_k) \bar{Q}_k\right]^{-1} \Rightarrow M_k = \nabla^2 f(\underline{x}_k)$

- If $\bar{Q}_k^T \nabla^2 f(\underline{x}_k) \bar{Q}_k$ is not PD, make it PD via cholesky decomposition
- It is possible to use Quasi-Newton versions for updating approximations of $\nabla^2 h(\underline{a}_k)$ or $\left[\nabla^2 h(\underline{a}_k)\right]^{-1}$

A General Projection Algorithm - 1

- Algorithm

1. Get initial $\underline{x}_0$ and working set $\mathcal{A}(\underline{x}_0)$. k=0
2. Determine $\underline{d}_k$
3. Determine maximum step size $\bar{\alpha}$ w/o violating any inactive constraints

$$\text{Find } \bar{\alpha} = \begin{cases} \dfrac{b_i - \underline{a}_i^T \underline{x}_k}{\underline{a}_i^T \underline{d}_k} & i \notin \mathcal{A}(\underline{x}_0) \ \& \ \underline{a}_i^T \underline{d}_k > 0 \\ \infty \ \text{ if for all } & i \notin \mathcal{A}(\underline{x}_0), \ \underline{a}_i^T \underline{d}_k < 0 \end{cases}$$

Use QR decomposition for numerical stability

4. Find $i_a = \arg\min_i \left(\dfrac{b_i - \underline{a}_i^T \underline{x}_k}{\underline{a}_i^T \underline{d}_k} \right), \ i \notin \mathcal{A}(\underline{x}_0) \ \& \ \underline{a}_i^T \underline{d}_k > 0$

$$\alpha_k = \arg\min_{\alpha \in (0,\bar{\alpha})} f\left(\underline{x}_k + \alpha_k \underline{d}_k\right)$$

5. If $\alpha_k < \bar{\alpha}$

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$$

A General Projection Algorithm - 2

If $\| \underline{d}_k \| < \varepsilon$ and $\mu_{i_d} = \min_i(\mu_i) > 0$ Stop

If $\mu_{i_d} < 0$, drop constraint i_d

$$A_{k+1} = A_k \,/\, \text{row } i_d \Rightarrow \text{ remove row } i_d \text{ from } A_k$$

Go to step 2

6. If $\alpha_k = \bar{\alpha}$, add constraint i_a to the set

$$A_{k+1} = A_k \cup \text{row } i_a \Rightarrow \text{ add row } i_a \text{ to } A_k$$

Go to step 2

- Linear Programming application

$$\min \underline{c}^T \underline{x}$$

$$\text{s.t.} \quad A\underline{x} = \underline{b}, \qquad \underline{x} \geq 0$$

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$$

$$\underline{d}_k = \arg\min \underline{c}^T \underline{d}_k + \frac{1}{2} \underline{d}_k^T M_k \underline{d}_k$$

$$\text{s.t.} \quad A\underline{d}_k = \underline{0}$$

Projection Applied to LP

$$\underline{d}_k = -M_k^{-1}\left(\underline{c} - A^T\underline{\lambda}_k\right)$$

$$\underline{\lambda}_k = \left(AM_k^{-1}A^T\right)^{-1} AM_k^{-1}\underline{c}$$

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k M_k^{-1}\left(\underline{c} - A^T\underline{\lambda}_k\right)$$

where

$$\alpha_k = \max\left\{\alpha \mid \underline{x}_k - \alpha M_k^{-1}\left(\underline{c} - A^T\underline{\lambda}_k\right) \geq 0\right\}$$

$$M_k = \left[\mathrm{Diag}\left\{x_{ki}\right\}\right]^{-2} = \left(X_k\right)^{-2}$$

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k X_k^2\left(\underline{c} - A^T\underline{\lambda}_k\right)$$

$$\underline{\lambda}_k = \left(AX_k^2A^T\right)^{-1} AX_k^2\underline{c}$$

- Affine scaling method due to Dikin (1967)
- Actually superior to Karmarkar's original method
- These methods belong to the so-called interior point methods for LP

Barrier (Penalty) Viewpoint of LP - 1

- More general view of LP via Barrier function methods

$$\min \underline{c}^T \underline{x}$$
$$\text{s.t.} \quad \underline{x} \in \Omega$$
$$\Omega = \{\underline{x} \mid \underline{x} \geq 0, A\underline{x} = \underline{b}\}$$

$$f_\varepsilon(\underline{x}) = \underline{c}^T \underline{x} - \varepsilon \sum_{i=1}^{n} \ln x_i$$
$$\text{s.t.} \quad A\underline{x} = \underline{b}$$

$$\underline{g}_k = \underline{c} - \varepsilon X_k^{-1} \underline{e} \qquad X_k = \text{diag}\left(x_{ki}\right)$$

$$\nabla^2 f_\varepsilon = \varepsilon X_k^{-2}$$

$$\underline{d}_k = \arg\min_{\underline{d}} \underline{g}_k^T \underline{d} + \frac{1}{2} \underline{d}^T \nabla^2 f_\varepsilon \underline{d} \qquad \text{Newton direction}$$
$$\text{s.t.} \quad A\underline{d} = \underline{0}$$

$$\Rightarrow \underline{d}_k = -\frac{1}{\varepsilon} X_k^2 \left(c - \varepsilon X_k^{-1} \underline{e} - A^T \underline{\lambda}_k\right)$$

Barrier (Penalty) Viewpoint of LP - 2

$$\underline{\lambda}_k = \left(AX_k^2A^T\right)^{-1} AX_k^2\left(\underline{c} - \varepsilon X_k^{-1}\underline{e}\right)$$

$$= \underbrace{\left(AX_k^2A^T\right)^{-1} AX_k^2\underline{c}}_{\underline{\lambda}_a} - \varepsilon \underbrace{\left(AX_k^2A^T\right)^{-1} AX_k\underline{e}}_{\underline{\lambda}_c}$$

$$\underline{d}_k = -\frac{1}{\varepsilon} X_k^2\left(\underline{c} - \varepsilon X_k^{-1}\underline{e} - A^T\left(\underline{\lambda}_a - \underline{\lambda}_c\varepsilon\right)\right)$$

$$\underline{d}_k = +\frac{1}{\varepsilon}\underline{d}_a - \underline{d}_c$$

$$\underline{d}_a = -X^2\left(c - A^T\underline{\lambda}_a\right)$$

$$\underline{d}_c = -X\underline{e} + X^2A^T\underline{\lambda}_c$$

So, Newton direction is a combination of affine scaling direction and so called centering direction that minimizes

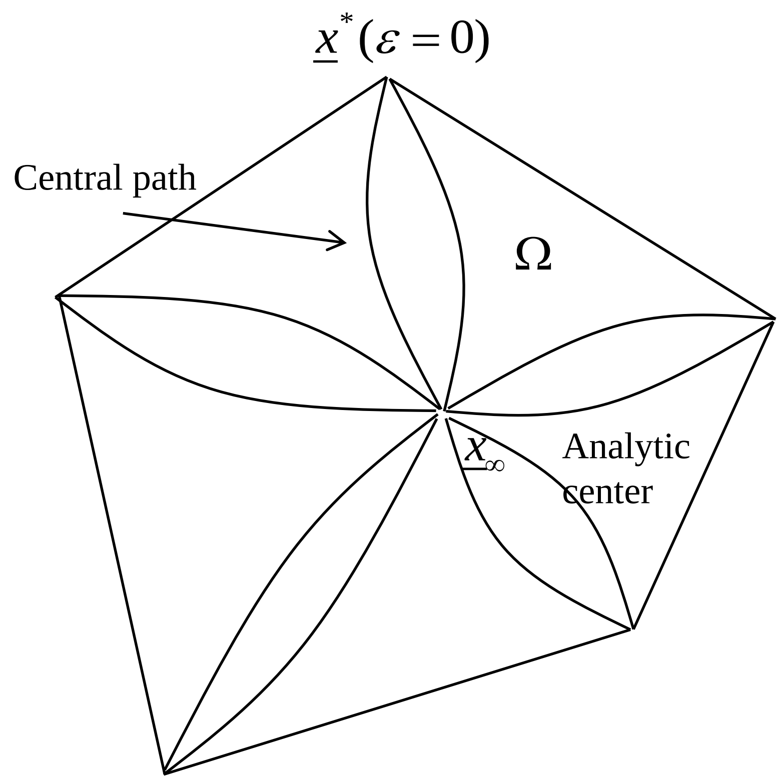
$$f_c = -\sum_{i=1}^{n} \ln x_i$$

$$\text{s.t. } \underline{x} \in \Omega = \left\{\underline{x} \mid A\underline{x} = \underline{b};\ \underline{x} \ge 0\right\}$$

Barrier (Penalty) Viewpoint of LP - 3

$$\underline{g}_c = -X^{-1}\underline{e}, \qquad \nabla^2 f_c = X^{-2}$$

$$\Rightarrow \underline{d}_c = -X\underline{e} + X^2 A^T \underline{\lambda}_c$$



$$-X^{-1}\underline{e} + A^T\underline{\lambda} = 0$$

$$A\underline{x} = \underline{b}$$

$$X^{-1}\underline{e} = A^T\underline{\lambda}$$

Sequential QR Updates - 1

- Updating QR decompositions:
  - QR decomposition for initial working set $\mathcal{A}(\underline{x}_0)$ from
    - Householder Reflections
    - (or) Gram-Schmidt (parallel)
    - (or) Givens Rotations
  - Each add (or drop) constraint modifies A^T in one column only. The procedure for modifying QR decompositions is as follows:

Suppose we have

$$QR = A^T = [\underline{a}_1\ \underline{a}_2 \ldots\ldots \underline{a}_{m+r}] = [Q_1\ \bar{Q}_1]\begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R$$

Suppose we want to find QRdecomposition of $\tilde{A}^T = [\underline{a}_1\ \underline{a}_2 .. \underline{a}_{i-1} \underline{a}_{i+1} \ldots\ \underline{a}_{m+r}]$

Split R as:

$$\begin{bmatrix} R_{11} & v & R_{13} \\ 0 & r_{ii} & w^T \\ 0 & 0 & R_{33} \end{bmatrix} \text{ so } Q^T \tilde{A}^T = \begin{bmatrix} R_{11} & R_{13} \\ 0 & \underline{w}^T \\ 0 & R_{33} \end{bmatrix}$$

H upper Hessenberg in the last $m - i$ columns

Sequential QR Updates - 2

Use Givens rotations

$$J_{m-1}^T \ldots\ldots J_{i+1}^T J_i^T H = \tilde{R}$$

$$\Rightarrow \tilde{Q} = Q J_i J_{i+1} \ldots J_{m-1}$$

Suppose want to add a column

$$\tilde{A}^T = [\underline{a}_1 \underline{a}_2 \ldots. \underline{a}_{m+r} \underline{a}_{m+r+1}] = [A^T \ \underline{a}_{i_a}]$$

$$Q^T \tilde{A}^T = \begin{bmatrix} R & Q^T \underline{a}_{i_a} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R & \underline{w} \\ 0 & 1 \end{bmatrix}; J_{m+r+1}^T \ldots. J_{n-1}^T \underline{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{m+r+1} \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\tilde{Q} = Q J_{n-1} \ldots\ldots J_{m+r+1}$$

– <u>Convergence Analysis</u>: Rate of convergence depends on Eigen values of $P_k \nabla^2 f(\underline{x}_k) P_k$ where P_k is the projection matrix associated with active constraints. For ROSEN's gradient projection method

$$\lim_{k \to \infty} \frac{f(\underline{x}_{k+1}) - f(\underline{x}^*)}{f(\underline{x}_k) - f(\underline{x}^*)} \leq \left(\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} \right)^2$$

Optimization with Simple Constraints - 1

For reduced gradient method, convergence rate depends on eigen values of

$$\left(\bar{Q}_k\bar{Q}_k^T\right)^{1/2}\nabla^2 f\left(\underline{x}^*\right)\left(\bar{Q}_k\bar{Q}_k^T\right)^{1/2}$$

For Newton type methods,convergence rate depends on eigen values of

$$\lim_{k\to\infty}\left[\bar{Q}_k\left[\bar{Q}_k^T\nabla^2 f\left(\underline{x}_k\right)\bar{Q}_k\right]^{-1}\bar{Q}_k^T\right]^{1/2}\nabla^2 f\left(\underline{x}_k\right)\left\{\bar{Q}_k\left[\bar{Q}_k^T\nabla^2 f\left(\underline{x}_k\right)\bar{Q}_k^T\right]^{-1}\bar{Q}_k^T\right\}^{1/2}$$

- Optimization with simple constraints

$$\min \; f(\underline{x})$$

$$\text{s.t.} \quad \underline{x} \geq \underline{0}$$

Optimality conditions:

$$\frac{\partial f}{\partial x_i}=0 \quad \text{if } x_i^*>0; \frac{\partial f}{\partial x_i}\geq 0 \quad \text{if } x_i^*=0$$

$$\Rightarrow \underline{x}^*=\left[\underline{x}^*-\alpha\nabla f\left(\underline{x}^*\right)\right]^+ ;\text{where} \qquad \underline{z}^+=\begin{bmatrix}\max\left(0,z_1\right)\\ \max\left(0,z_2\right)\\ \vdots\\ \max\left(0,z_n\right)\end{bmatrix}$$

Optimization with Simple Constraints - 2

– <u>A Steepest descent method:</u>

$$\underline{x}_{k+1} = \left[\underline{x}_k - \alpha_k \nabla f\left(\underline{x}_k\right)\right]^+; \quad k = 0,1,2,\ldots..$$

$$\text{where} \quad \alpha_k = \arg\min_{\alpha} f\left(\left[\underline{x}_k - \alpha \nabla f\left(\underline{x}_k\right)\right]^+\right); \alpha \geq 0$$

– <u>Generalized gradient method:</u>

$$\underline{x}_{k+1} = \left[\underline{x}_k - \alpha_k H_k \nabla \underline{f}\left(\underline{x}_k\right)\right]^+$$

$$\text{where} \quad \alpha_k = \arg\min_{\alpha} f\left(\left[\underline{x}_k - \alpha H_k \nabla \underline{f}\left(\underline{x}_k\right)\right]^+\right); \alpha \geq 0$$

$$\text{Typically, } H_k \text{ is selected to be diagonal } H_k = \begin{bmatrix} h_1 & & & \\ & h_2 & & \\ & & \vdots & \\ & & & h_n \end{bmatrix}; h_i = \left[\frac{\partial^2 f\left(\underline{x}_k\right)}{\partial x_i}\right]^{-1}$$

– <u>Extension to upper and lower bound constraints:</u>

$$\underline{b}_1 \leq \underline{x} \leq \underline{b}_2$$

$$\underline{x}_{k+1} = \left[\underline{x}_k - \alpha_k H_k \nabla \underline{f}\left(\underline{x}_k\right)\right]^\# \text{ where } \left[\underline{z}\right]^\# \text{ is given by} \quad z_i^\# = \begin{cases} b_{2i} \text{ if } b_{2i} \leq z_i \\ z_i \text{ if } b_{1i} < z_i < b_{2i} \\ b_{1i} \text{ if } z_i \leq b_{1i} \end{cases}$$

Optimization with Simple Constraints - 3

– Combining with Projected Newton's method:

Suppose at the first trail point $\left[\underline{x}_k - \alpha H_k \nabla \underline{f}\left(\underline{x}_k\right)\right]^+ = \underline{x}_k\left(\alpha\right)$

the set of active constraints $A\left(\underline{x}_k\left(\alpha\right)\right) \neq A\left(\underline{x}_k\right)$; where

$$A\left(\underline{x}_k\left(\alpha\right)\right) = \left\{ i \,\middle|\, 0 \leq x_{ki}\left(\alpha\right) \leq \varepsilon_k, \frac{\partial f\left(x_k\left(\alpha\right)\right)}{\partial x_i} > 0 \right\}; \varepsilon_k \text{ is small}$$

Then, we use Diagonally scaled steepest descent.

If $A\left(\underline{x}_k\left(\alpha\right)\right) = A\left(\underline{x}_k\right)$, use projected Newton, since probably close to optimal.

What is the form of projected Newton?

Projected Hessian

$$F_p\left(\underline{x}_k\right) = \begin{cases} \dfrac{\partial^2 f}{\partial x_i \partial x_j} \\ 0 \text{ if } i \neq j \text{ and either } i \in A\left(\underline{x}_k\right) \text{ or } j \in A\left(\underline{x}_k\right) \end{cases}$$

[It does not matter what you set diagonals of $i \in A\left(\underline{x}_k\right)$ and $i = j$]

Optimization with Simple Constraints - 4

Projected gradient

$$\underline{g}_p\left(\underline{x}_k\right)=\begin{cases}\dfrac{\partial f\left(\underline{x}_k\right)}{\partial x_i} & \text{if } i\notin A\left(\underline{x}_k\right)\\ 0 & \text{otherwise}\end{cases}$$

Use $H_k\underline{g}_p\left(\underline{x}_k\right)=\left[F_p\left(\underline{x}_k\right)\right]^{-1}\underline{g}_p\left(\underline{x}_k\right)$

- Note directions are zero for active constraints as they should.
  - Extension to upper and lower bound constraints :

    $\underline{b}_1\leq\underline{x}\leq\underline{b}_2$

    Same as before except

$$A\left(\underline{x}_k\right)=\left\{i\,\middle|\,b_{1i}\leq x_{ki}\leq b_{1i}+\varepsilon_k \text{ and } \partial f\left(\underline{x}_k\right)\big/\partial x_i>0\,(\text{or})\right.$$

$$\left.b_{2i}-\varepsilon_k\leq x_{ki}\leq b_{2i} \text{ and } \partial f\left(\underline{x}_k\right)\big/\partial x_i<0\right\}$$

Optimization with Discerete Variables - 1

– Consider the problem

$$\min_{\underline{x}\in X} f(\underline{x})$$

$$s.t.\ \underline{g}(\underline{x}) \le \underline{0}$$

Lagrangian $L(\underline{x},\underline{\mu}) = f(\underline{x}) + \underline{\mu}^T \underline{g}(\underline{x})$

$$q(\underline{\mu}) = \min_{\underline{x}\in X} L(\underline{x},\underline{\mu}) = \min_{\underline{x}\in X} \left\{ f(\underline{x}) + \underline{\mu}^T \underline{g}(\underline{x}) \right\} \Rightarrow \underline{x}_{\mu}$$

$\underline{g}(\underline{x}_{\mu})$ is a subgradient of the dual function q at $\underline{\mu}$

$$q(\overline{\underline{\mu}}) \le q(\underline{\mu}) + (\overline{\underline{\mu}} - \underline{\mu})^T \underline{g}(\underline{x}_{\mu})\ \forall \overline{\underline{\mu}} \in R^r$$

$Why?$

$$q(\overline{\underline{\mu}}) = \min_{\underline{x}\in X}\{ f(\underline{x}) + \overline{\underline{\mu}}^T \underline{g}(\underline{x})\} \le f(\underline{x}_{\mu}) + \overline{\underline{\mu}}^T \underline{g}(\underline{x}_{\mu})$$

$$= f(\underline{x}_{\mu}) + \underline{\mu}^T \underline{g}(\underline{x}_{\mu}) + (\overline{\underline{\mu}} - \underline{\mu})^T \underline{g}(\underline{x}_{\mu}) = q(\underline{\mu}) + (\overline{\underline{\mu}} - \underline{\mu})^T \underline{g}(\underline{x}_{\mu})$$

When we get $\underline{x}_{\mu}$, the subgradient is obtained at no cost.

When $q(\underline{\mu})$ is differentiable, subgradient is equivalent to a gradient

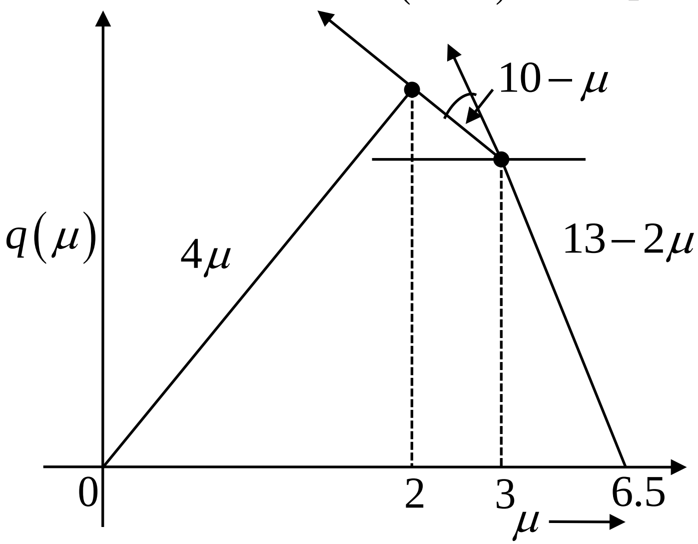
Example

- Example: $\min_{x_1,x_2} \; 10x_1 + 3x_2$

$$s.t. \quad 5x_1 + x_2 \ge 4$$
$$x_1, x_2 = 0 \text{ or } 1$$

$$L(\underline{x},\mu) = 10x_1 + 3x_2 + \mu(4 - 5x_1 - x_2)$$

$$q(\mu) = \begin{cases} 4\mu & \text{if } \mu \le 2 \\ 10-\mu & \text{if } 2 \le \mu \le 3 \\ 13-2\mu & \text{if } \mu \ge 3 \end{cases}$$



$\partial q(\mu) = g(\mu)$ is the convex hull of: $-\xi_1, -2\xi_2$

$\ni \xi_1 + \xi_2 = 1,\; \xi_1, \xi_2 \ge 0$ at $\mu = 3$

$\Rightarrow -\xi_1 - 2(1-\xi_1) = \xi_1 - 2 \in (-2,-1)$

$$\underline{\mu}^{k+1} = \left[\underline{\mu}^k + s^k \; \underline{g}^k\right]^+ \qquad [x]_i^+ = \begin{cases} x_i \text{ if } x_i \ge 0 \\ 0 \text{ otherwise} \end{cases}$$

In constrast to the regular gradient method, the new iterate may not improve the dual cost,i.e., we may have $q\left(\left[\underline{\mu}_k + s^k \underline{g}^k\right]^+\right) < q(\underline{\mu}_k) \forall s > 0$

Convergence Property: For $s^k \ni 0 < s^k < \dfrac{2\left[q(\underline{\mu}^*) - q(\underline{\mu}^k)\right]}{\left\|\underline{g}^k\right\|^2}$

$$\left\|\underline{\mu}^{k+1} - \underline{\mu}^*\right\| < \left\|\underline{\mu}^k - \underline{\mu}^*\right\|$$

Proof of Convergence Property

- Proof: $\left\|\underline{\mu}^k + s^k \underline{g}^k - \underline{\mu}^*\right\|^2 = \left\|\underline{\mu}^k - \underline{\mu}^*\right\|^2 - 2s^k \left(\underline{\mu}^* - \underline{\mu}^k\right)^T \underline{g}^k + (s^k)^2 \left\|\underline{g}^k\right\|^2$

  also $q\left(\underline{\mu}^*\right) \le q\left(\underline{\mu}^k\right) + \left(\underline{\mu}^* - \underline{\mu}^k\right)^T \underline{g}^k$

  $\Rightarrow \left(\underline{\mu}^* - \underline{\mu}^k\right)^T \underline{g}^k \ge q\left(\underline{\mu}^*\right) - q\left(\underline{\mu}^k\right)$

  $\Rightarrow \left\|\underline{\mu}^k + s^k \underline{g}^k - \underline{\mu}^*\right\|^2 \le \left\|\underline{\mu}^k - \underline{\mu}^*\right\|^2 - 2s^k \left[q\left(\underline{\mu}^*\right) - q\left(\underline{\mu}^k\right)\right] + (s^k)^2 \left\|\underline{g}^k\right\|^2$

  $$= \left\|\underline{\mu}^k - \underline{\mu}^*\right\|^2 - \gamma^k\left(2 - \gamma^k\right)\frac{\left(q\left(\underline{\mu}^*\right) - q\left(\underline{\mu}^k\right)\right)^2}{\left\|\underline{g}^k\right\|^2}$$

  where $\gamma^k = \dfrac{s^k \left\|\underline{g}^k\right\|^2}{q\left(\underline{\mu}^*\right) - q\left(\underline{\mu}^k\right)}$

  so, if $0 < \gamma^k < 2, \left\|\underline{\mu}^k + s^k \underline{g}^k - \underline{\mu}^*\right\|^2 \le \left\|\underline{\mu}^* - \underline{\mu}^k\right\|^2 \Rightarrow \underline{\mu}^k + s^k \underline{g}^k$ is closer to $\underline{\mu}^*$

  The projection operator reduces the left hand side even further.

- How to select s^k: Do not know $q\left(\underline{\mu}^*\right)$

  $$s^k = \frac{\alpha^k \left(q^k - q\left(\underline{\mu}^k\right)\right)}{\left\|\underline{g}^k\right\|^2}$$

  q^k is an approximation to q^* and $0 < \alpha^* < 2$

Step Size Selection

- Choices

1) $q^k = \max_{0 \le i \le k} q\left(\underline{\mu}^i\right)$

2) Best feasible cost $f\left(\bar{x}\right)$

   – Initially $\alpha^k = 1$ and is decreased by a factor of two every few iterations (e.g. , 5 or 10).

   – $\alpha^k = \dfrac{1+m}{k+m}$

3) $\alpha^k = 1$ for all k

$$q^k = \left(1+\beta(k)\right)\hat{q}^k; \hat{q}^k = \max_{0 \le i \le k} q\left(\underline{\mu}^i\right); \beta(k) = \begin{cases} \beta(k)\gamma & \text{if success } \gamma > 1 \\ \beta(k)\delta & \text{if failure } \delta < 1 \end{cases}$$

❑ Advantages:

- Ideas are intuitively clear
- Straightforward implementation
- Computationally efficient
- Works well

❑ Disadvantages: Not easy to tune the parameters α and β

Cutting plane Methods

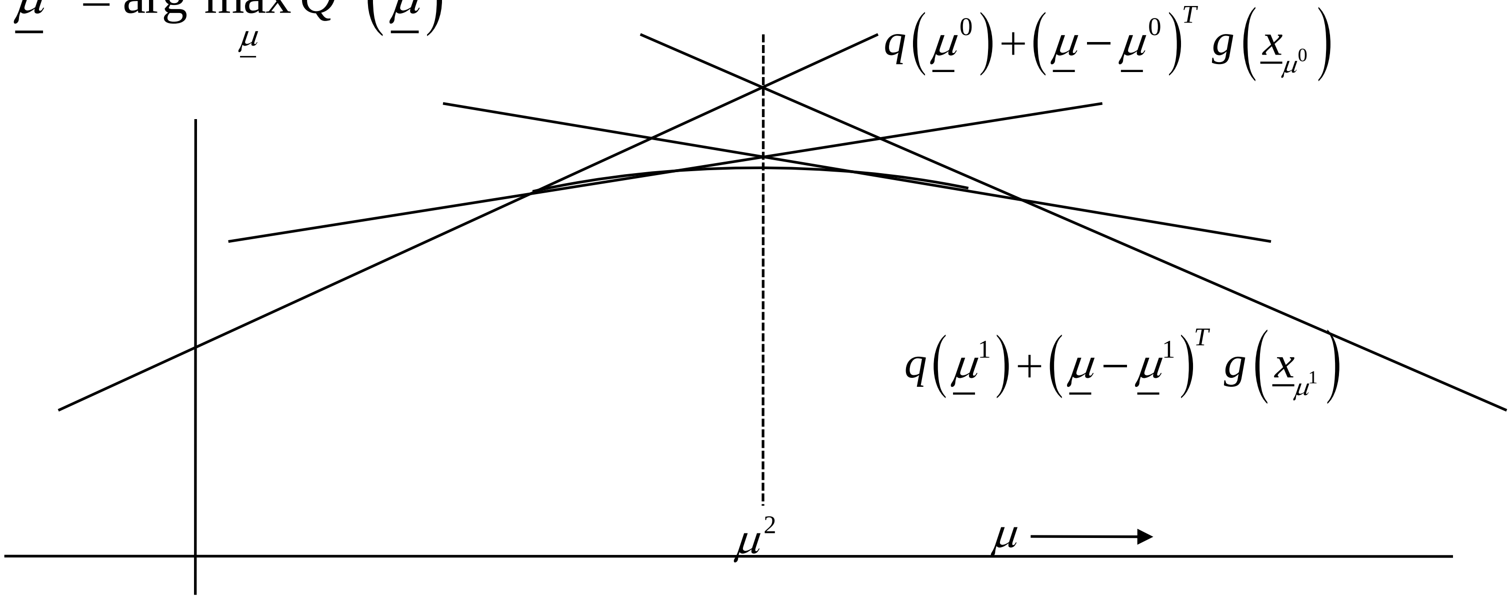
- Cutting plane methods

$$\max_{\underline{\mu} \geq \underline{0}} q\left(\underline{\mu}\right)$$

$$\max_{\underline{\mu}} \; Q^k\left(\underline{\mu}\right) \quad \text{s.t. } \underline{\mu} \geq \underline{0}$$

$$Q^k\left(\underline{\mu}\right) = \min\left\{q\left(\underline{\mu}^0\right) + \left(\underline{\mu} - \underline{\mu}^0\right)^T \underline{g}^0, \ldots\ldots, q\left(\underline{\mu}^{k-1}\right) + \left(\underline{\mu} - \underline{\mu}^{k-1}\right)^T \underline{g}^{k-1}\right\}$$

$$\underline{\mu}^k = \arg\max_{\underline{\mu}} Q^k\left(\underline{\mu}\right)$$



Summary

- Review: Multiplier (primal-dual and augmented Lagrangian methods) and Successive Quadratic Programming methods
- Feasible Direction Methods
  - Rosen's Gradient Projection methods
  - Reduced Gradient method
  - Newton-type methods
  - Updating QR decompositions
- Problems with Simple Constraints
- Subgradient Methods for Discrete Optimization
- Cutting plane methods