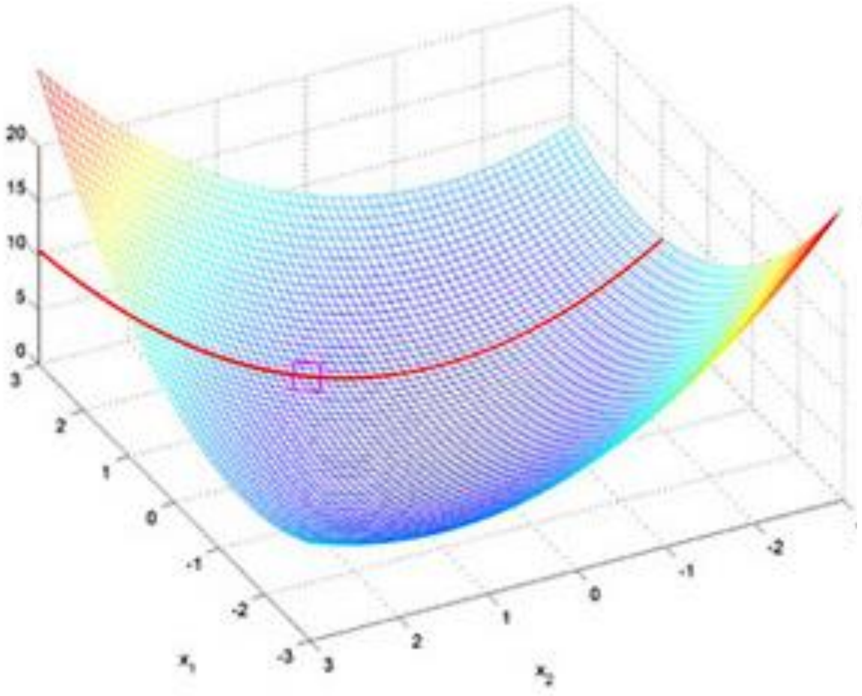
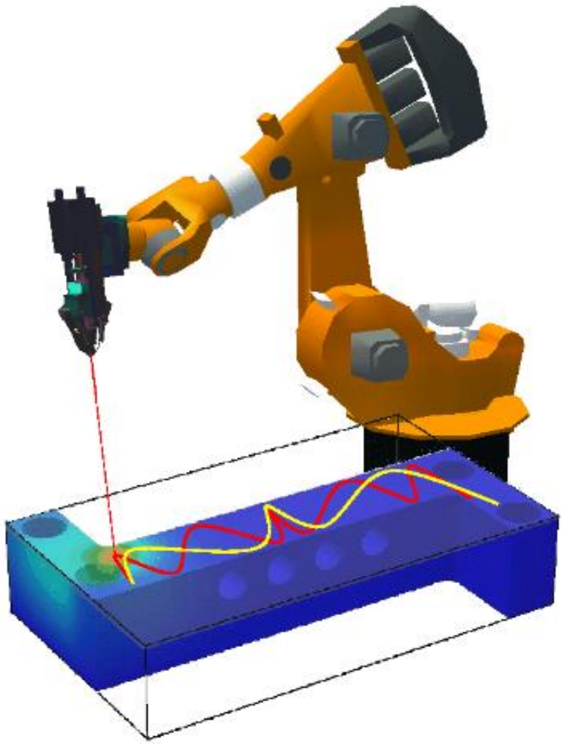
MEGR 3090/7090/8090: Advanced Optimal Control



$$\boxed{V_n(\mathbf{x}_n) = \min_{\{\mathbf{u}_n,\mathbf{u}_{n+1},\cdots,\mathbf{u}_{N-1}\}} \left[\frac{1}{2}\sum_{k=n}^{N-1}\left(\mathbf{x}_k^T\mathbf{Q}_k\mathbf{x}_k + \mathbf{u}_k^T\mathbf{R}\mathbf{u}_k\right) + \frac{1}{2}\mathbf{x}_N^T\mathbf{Q}_N\mathbf{x}_N \right]}$$

$$\begin{aligned}
V_n(\mathbf{x}_n) &= \min_{\{\mathbf{u}_n,\mathbf{u}_{n+1},\cdots,\mathbf{u}_{N-1}\}} \left[\frac{1}{2}\sum_{k=n}^{N-1}\left(\mathbf{x}_k^T\mathbf{Q}_k\mathbf{x}_k + \mathbf{u}_k^T\mathbf{R}\mathbf{u}_k\right) + \frac{1}{2}\mathbf{x}_N^T\mathbf{Q}_N\mathbf{x}_N \right] \\
&= \min_{\mathbf{u}_n} \left[\frac{1}{2}\left(\mathbf{x}_n^T\mathbf{Q}_n\mathbf{x}_n + \mathbf{u}_n^T\mathbf{R}\mathbf{u}_n\right) + \underbrace{\min_{\{\mathbf{u}_{n+1},\cdots,\mathbf{u}_{N-1}\}} \left[\frac{1}{2}\sum_{k=n+1}^{N-1}\left(\mathbf{x}_k^T\mathbf{Q}_k\mathbf{x}_k + \mathbf{u}_k^T\mathbf{R}\mathbf{u}_k\right) + \frac{1}{2}\mathbf{x}_N^T\mathbf{Q}_N\mathbf{x}_N \right]}_{V_{n+1}(\mathbf{x}_{n+1})} \right] \\
&= \min_{\mathbf{u}_n} \left[\frac{1}{2}\left(\mathbf{x}_n^T\mathbf{Q}_n\mathbf{x}_n + \mathbf{u}_n^T\mathbf{R}\mathbf{u}_n\right) + V_{n+1}(\mathbf{x}_{n+1}) \right]
\end{aligned}$$

$$\boxed{V_n(\mathbf{x}_n) = \min_{\mathbf{u}_n} \left[\frac{1}{2}\left(\mathbf{x}_n^T\mathbf{Q}_n\mathbf{x}_n + \mathbf{u}_n^T\mathbf{R}\mathbf{u}_n\right) + V_{n+1}(\mathbf{x}_{n+1}) \right]}$$



Lecture 11

September 26, 2017

A Note About Unconstrained LQR

Suppose we want to minimize $J(\underline{u}; \underline{x}(0)) = \sum_{i=0}^{N-1} [\underline{x}(i+1)^T Q \underline{x}(i+1) + R u(i)^2]$

Given: $\underline{x}(k+1) = A\underline{x}(k) + Bu(k)$

Fun fact: As $N \to \infty$, $u^*(k)$ can be represented by $u^*(k) = -K\underline{x}(k)$ for some K.

Quadratic Design and Control Optimization Problems - Reminder

UNC CHARLOTTE

Quadratic programming (QP) optimization problem:

Minimize: $$J(\mathbf{u}) = \mathbf{u}^T Q\mathbf{u} + R\mathbf{u}$$

Subject to: $$A_1\mathbf{u} - \mathbf{b}_1 \leq \mathbf{0}$$
$$A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$$

Linear quadratic regulator (LQR) problem:

Minimize: $$J(\mathbf{u}, \mathbf{x}_0) = \sum_{i=0}^{N-1} (\mathbf{x}(i)^T Q\mathbf{x}(i) + Ru(i)^2)$$

Subject to: $$Mu(i) - b \leq 0, i = 0 \dots N-1$$
$$\mathbf{x}(i+1) = A\mathbf{x}(i) + Bu(i)$$

Quadratic Design and Control Optimization Problems - Reminder

UNC CHARLOTTE

QP formulation: Minimize $J(\underline{u}) = \underline{u}^T Q \underline{u} + R\underline{u}$

Subject to: $A_1 \underline{u} - \underline{b}_1 \leq 0$

$A_2 \underline{u} - \underline{b}_2 = 0$

LQR problem formulation:

Minimize $J(\underline{u}; \underline{x}_0) = \sum_{i=0}^{N-1} [\underline{x}(i+1)^T Q \underline{x}(i+1) + Ru(i)^T]$

Subject to: $Mu(i) - b \leq 0, \quad i = 0 \ldots N-1$

$\underline{x}(i+1) = A\underline{x}(i) + Bu(i)$

$\Rightarrow J(\underline{u}; \underline{x}_0) = \underline{u}^T \bar{Q} \underline{u} + \underline{c}^T \underline{u} + k$ (HW 1, prob. 4)

Minimizing $\underline{u}^T \bar{Q} \underline{u} + \underline{c}^T \underline{u} + k$ is equiv. to minimizing

$\underline{u}^T \bar{Q} \underline{u} + \underline{c}^T \underline{u}$

$\underbrace{M I_{N\times N}}_{\text{"}A_1\text{"}} \underline{u} - \underbrace{b \underline{1}_{N\times 1}}_{\underline{b}_1} \leq \underline{0}$

KKT Conditions for Quadratic Programs – Assessment (Reminder)

Optimal control problem: Minimize: $J(\mathbf{u}) = \mathbf{u}^T Q\mathbf{u} + \mathbf{r}^T \mathbf{u}$ ← p decision variables

Subject to:

$A_1\mathbf{u} - \mathbf{b}_1 \leq \mathbf{0}$ ← q inequality constraints

$A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$ ← r equality constraints

QP-specific KKT conditions: $2Q\mathbf{u} + \mathbf{r}^T + \boldsymbol{\mu}^T A_1 + \boldsymbol{\lambda}^T A_2 = \mathbf{0}$ ← p **linear** equations

$A_2\mathbf{u}^* = \mathbf{b}_2$ ← r **linear** equations

$\mu_i \geq 0, i = 1 \ldots q$

$\lambda_i \neq 0, i = 1 \ldots r$

$\mu_i(A_{1i}\mathbf{u}^* - b_{1i}) = 0, i = 1 \ldots q$ ← q **nonlinear** equations

(Note: A_{1i} = i^th row of A_1, b_{1i} = i^th element of $\mathbf{b}_1$)

Key observations (same as with LP):

- For every i, either $\mu_i = 0$ or $A_{1i}\mathbf{u}^* - b_{1i} = 0$
- If we just knew which of the above were true for each i, we'd have q linear equations

KKT Conditions for Quadratic Programs – Assessment (Reminder)

Lagrangian: $L(\underline{u}, \underline{\mu}, \underline{\lambda}) = J(\underline{u}; \underline{x}(0)) + \underline{\mu}^T g(\underline{u}) + \underline{\lambda}^T h(\underline{u})$

1^{st} KKT condition: $\nabla_{\underline{u}} L(\underline{u}^*, \underline{\mu}, \underline{\lambda}) = \underline{0}$

Remember: The first KKT condition involves taking the derivative of the Lagrangian (with respect to u) and setting it equal to 0.

Challenges with Active Set Methods for Quadratic Programming

UNC CHARLOTTE

Challenge 1 (easy to get around) – Optimal point could be on the **interior or boundary** of $G_1 \cap G_2$
…Simple remedy: Do an unconstrained optimization first, and if $\mathbf{u}^*_{unconstrained}$ violates constraints, then some constraints must be active at $\mathbf{u}^*$

ACTIVE SET ALGORITHM

1. Input initial feasible point and working set.
2. Termination test (including KKT test). If the point is not optimal, either continue with same working set or go to 7.
3. Compute a feasible search vector $\mathbf{s}_k$.
4. Compute a step length α_k along $\mathbf{s}_k$, such that $f(\mathbf{x}_k + \alpha_k \mathbf{s}_k) < f(\mathbf{x}_k)$. If α_k violates a constraint, continue; otherwise go to 6.
5. Add a violated constraint to the constraint set and reduce α_k to the maximum possible value that retains feasibility.
6. Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{s}_k$.
7. Change the working set (if necessary) by deleting a constraint, update all quantities, and go to step 2.

Challenge 2 (more cumbersome) – Optimal point will not generally lie on a **vertex** of $G_1 \cap G_2$, so the simplex algorithm isn't applicable

Challenges with Active Set Methods for Quadratic Programming

Dealing with inequality constraints :
hard

Issues :

1) Always result in nasty complementary slackness condition.

2) Can result in infeasible optimization problem.

3) Leaving zero room for constraint violation can be a very safe thing (if the constraint exists for safety) but can also be very dangerous (if the constraint is there for performance).

Related to difficulty in applying KKT conditions, due to inequality

Due to "hardness"

Challenges with Active Set Methods for Quadratic Programming

Original optimization problem:

Minimize $J(\underline{u})$ $\quad \underline{u} \in \mathbb{R}^p$

Subj. to $g(\underline{u}) \leq 0$ $\quad g(\underline{u}) \in \mathbb{R}^q$

could be crazy complex

Modified problem:

Minimize $J(\underline{u})$, $\quad \underline{u} \in \mathbb{R}^p$

Subj. to: $g(\underline{u}) + \underline{s} = 0$, $\quad \underline{s} \in \mathbb{R}^q$

$\underline{s}$ = vector of slack variables

$s_i \geq 0, \forall i \leftarrow q$ constraints.

very simple.

Constraint Softening – Main Idea

If inequality constraints are causing us problems, we can replace them with **penalties**…

Original QP problem: Minimize: $J(\mathbf{u}) = \mathbf{u}^T Q\mathbf{u} + \mathbf{r}^T\mathbf{u}$

Subject to: $A_1\mathbf{u} - \mathbf{b}_1 \leq \mathbf{0}$
$A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$

New approximated QP problem: Minimize: $J(\mathbf{u}) = \mathbf{u}^T Q\mathbf{u} + \mathbf{r}^T\mathbf{u} + kP(\mathbf{u})$

Subject to: $A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$

- The **hard** inequality constraint from the original problem has been **softened**
- This can be done for more than just QP problems
- Constraint softening is a fundamental concept behind interior point methods

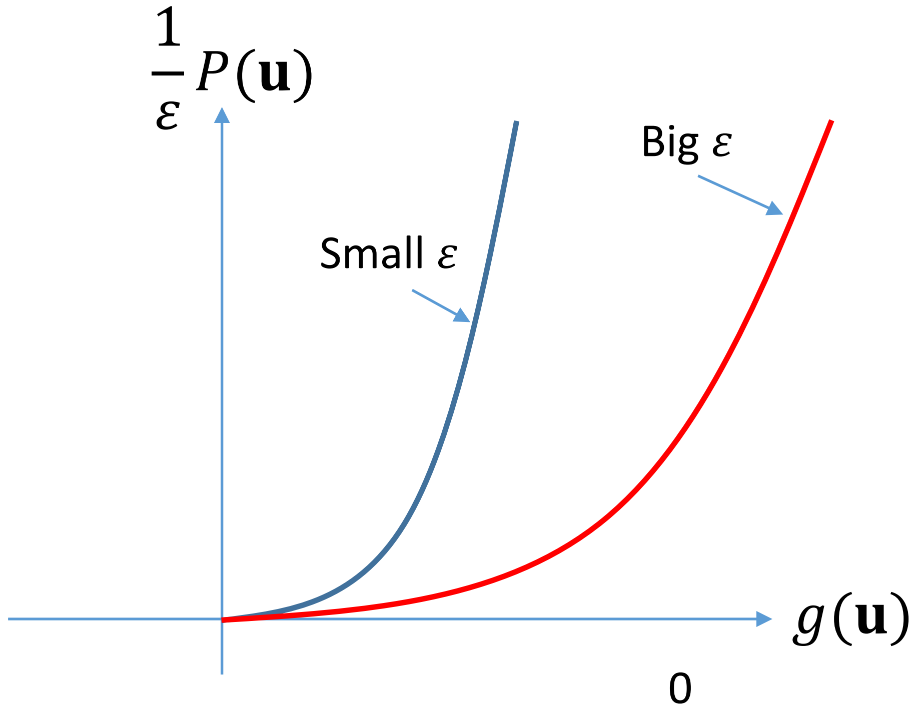
Constraint Softening via Penalty Functions

Approximated QP problem: Minimize: $J(\mathbf{u}) = \mathbf{u}^T Q \mathbf{u} + \mathbf{r}^T \mathbf{u} + \frac{1}{\varepsilon} P(\mathbf{u})$

Where: $P(\mathbf{u}) = [\max\{\mathbf{0}, \underbrace{A_1\mathbf{u} - \mathbf{b}_1}_{= g(\mathbf{u})}\}]^T K \max\{\mathbf{0}, \underbrace{A_1\mathbf{u} - \mathbf{b}_1}_{= g(\mathbf{u})}\}$

Subject to: $A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$

Visualization for a single constraint:



- Modified and original objective functions are identical when the inequality constraint is satisfied (i.e., when $g(\mathbf{u}) \leq 0$)
- Modified objective function increases rapidly when $g(\mathbf{u}) > 0$ (degree of rapidness depends on ε)

Constraint Softening via Penalty Functions

Idea: Modified optimization problem:

$K = kI$ (or diag. matrix)

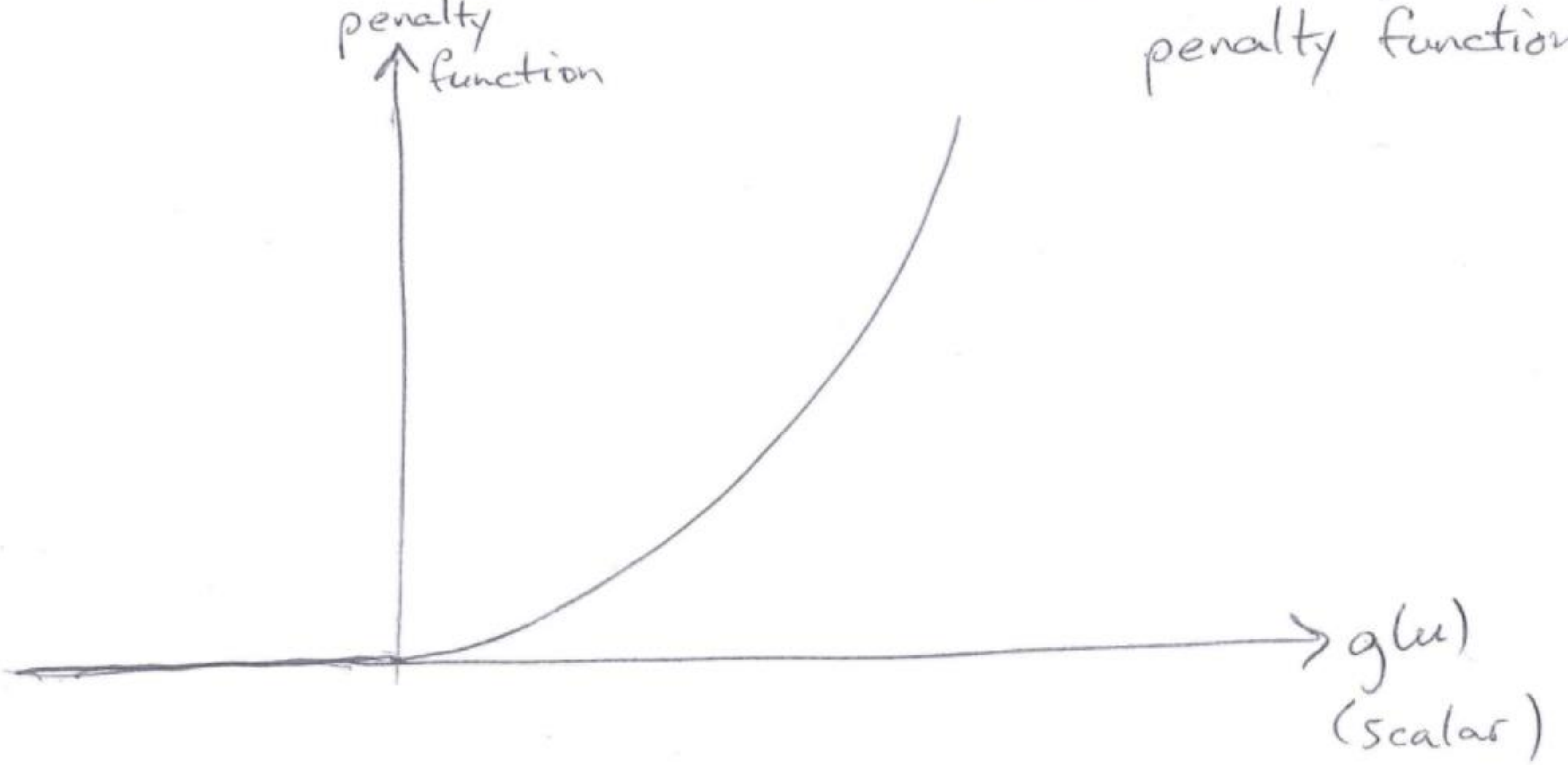
$$\text{Minimize } J_{mod}(\underline{u}) = J(\underline{u}) + \underbrace{\left(\min\{\underline{0}, \underline{s}\}\right)^T K \min\{\underline{0}, \underline{s}\}}_{\text{penalty function}}$$

Subject to $g(\underline{u}) + \underline{s} = 0$

Constraint Softening via Penalty Functions

Equivalent problem:

$$\text{Minimize } J_{mod}(\underline{u}) = J(\underline{u}) + \underbrace{(\max\{\underline{0}, g(\underline{u})\})^T K \max\{\underline{0}, g(\underline{u})\}}_{\text{penalty function}}$$



Constraint Softening via Barrier Functions

Approximated QP problem: Minimize: $J(\mathbf{u}) = \mathbf{u}^T Q\mathbf{u} + \mathbf{r}^T\mathbf{u} + \varepsilon B(\mathbf{u})$

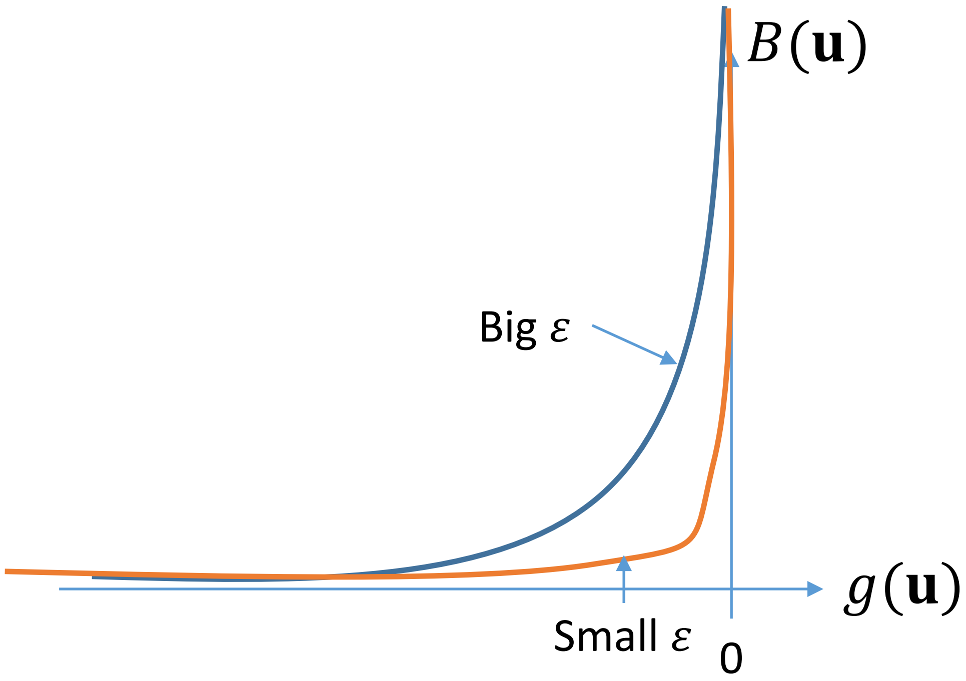
Where: $B(\mathbf{u}) = \sum_{i=1}^{q} -k_i \ln(\boxed{-A_{1i}\mathbf{u} + b_{1i}})$

$\triangleq -g_i(\mathbf{u})$

Subject to: $A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$

- Modified and original objective functions are similar far from the inequality constraint, on the feasible side of the constraint set
- Modified objective function goes to ∞ at the boundary of the constraint set (hence acting as a barrier)

Visualization for a single constraint:



Key Ideas of Interior Point Method

Starting point - Modified objective function (augmented with a barrier or penalty – barrier shown here):

Minimize: $J(\mathbf{u}) = \mathbf{u}^T Q\mathbf{u} + \mathbf{r}^T\mathbf{u} + \varepsilon B(\mathbf{u})$

Where: $B(\mathbf{u}) = \sum_{i=1}^{q} -k_i \ln(\underbrace{-A_{1i}\mathbf{u} + b_{1i}}_{\triangleq -g_i(\mathbf{u})})$

Subject to: $A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}$

Main optimization process:

- Search for the minimizer of the modified objective function above
- Compute the limit of $\mathbf{u}^*$ as $\varepsilon \longrightarrow 0$

Two ways to perform the optimization:

- Option 1 (not very common) – The analytical way…Write out equations for $\nabla J(\mathbf{u};\varepsilon)$, set $\nabla J(\mathbf{u};\varepsilon) = \mathbf{0}$, then solve for $\mathbf{u}$ in terms of ε… $\mathbf{u}^* = \lim_{\varepsilon \longrightarrow 0} \mathbf{u}\,(\varepsilon)$
- Option 2 (very common and available as a built-in MATLAB function – The numerical way…Start with a relatively large value of ε at iteration 1, then decrease ε at each iteration

Interior Point Method – Simple Analytical Example (Papalambros Ex. 7.7)

UNC CHARLOTTE

Minimize: $J(\mathbf{u}) = 3u_1^2 + u_2^2$

Subject to: $u_1 + u_2 \geq 2$

Main steps:

1. Augment $J(\mathbf{u})$ with a barrier function.
2. Compute $\nabla J(\mathbf{u}; \varepsilon)$.
3. Set $\nabla J(\mathbf{u}; \varepsilon) = 0$, and solve for $\mathbf{u}(\varepsilon)$.
4. Compute $\lim_{\varepsilon \longrightarrow 0} \mathbf{u}\,(\varepsilon)$

Interior Point Method – Simple Analytical Example (Papalambros Ex. 7.7)

UNC CHARLOTTE

Ex: $J(\underline{u}) = 3u_1^2 + u_2^2$

Subj. to: $u_1 + u_2 \geq 2$

$$J_{mod}(\underline{u}) = 3u_1^2 + u_2^2 - \varepsilon \ln(u_1 + u_2 - 2)$$

$$\nabla J_{mod} = [6u_1 - \varepsilon(u_1 + u_2 - 2)^{-1} \quad 2u_2 - \varepsilon(u_1 + u_2 - 2)^{-1}] = 0 \text{ when } \underline{u} = \underline{u}^*$$

$$6u_1^* - \varepsilon(u_1^* + u_2^* - 2)^{-1} = 0$$

$$2u_2^* - \varepsilon(u_1^* + u_2^* - 2)^{-1} = 0$$

$$\Rightarrow u_2^* = 3u_1^*$$

Interior Point Method – Simple Analytical Example (Papalambros Ex. 7.7)

$$6u_1^* - \varepsilon(u_1^* + u_2^* - 2)^{-1} = 0$$

$$\Rightarrow 6u_1^*(u_1^* + u_2^* - 2) - \varepsilon = 0$$

$$\Rightarrow \underbrace{6u_1^{*2} + 18u_1^{*2}}_{24u_1^{*2}} - 12u_1^* - \varepsilon = 0$$

$$\Rightarrow u_1^*(\varepsilon) = \frac{12 \pm \sqrt{144 + 96\varepsilon}}{48} = \frac{12 \pm 12\sqrt{1 + \frac{2}{3}\varepsilon}}{48}$$

$$= \frac{1}{4} \pm \frac{1}{4}\sqrt{1 + \frac{2}{3}\varepsilon}$$

$$u_1^* = \lim_{\varepsilon \to 0} u_1^*(\varepsilon) = \cancel{0} \text{ or } \frac{1}{2} = u_1^* \checkmark$$

$$\cancel{u_2^* = 0} \qquad u_2^* = 3/2 \checkmark$$

constraints violated

Interior Point Method – Numerical Process

Main optimization process:

- Search for the minimizer of the modified objective function above
- Compute the limit of $\mathbf{u}^*$ as $\varepsilon \longrightarrow 0$

Main idea of numerical optimization: Start with a relatively large value of ε at iteration 1, then decrease ε at each iteration

Syntax:

```
[u_opt,J_opt] = quadprog(Q,r,A1,b1,A2,b2);
```

$\mathbf{u}^*$ $\quad J(\mathbf{u}^*)$ $\quad Q$ $\quad \mathbf{r}$ $\quad A_1$ $\quad \mathbf{b}_1$ $\quad A_2$ $\quad \mathbf{b}_2$

https://www.mathworks.com/help/optim/ug/quadprog.html

Interior Point Method – Linear Quadratic Regulator Example

System dynamics:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + Bu(k)$$
$$y(k) = C\mathbf{x}(k)$$

where:

$$A = \begin{bmatrix} 0.99 & 0.1 \\ -0.2 & 0.9 \end{bmatrix}$$

$$B = [0 \quad 0.1]^T$$

$$C = [1 \quad 0]$$

Minimize: $J\big(u, \mathbf{x}(0)\big) = \sum_{i=1}^{5} \mathbf{x}^T(i)\bar{Q}\mathbf{x}(i) + \bar{R}u(i)^2$ where: $\bar{Q} = I_{2\times 2}, \bar{R} = 1$

Subject to system dynamics and: $-0.5 \le u(i) \le 0.5, i = 0 \ldots 4$ $\quad \mathbf{x}(0) = [5 \quad 0]^T$

Code available on Canvas

Preview of Upcoming Lectures

Next 2 lectures – Sequential quadratic programming

- Leads to a ***locally optimal*** solution for more complex optimization problems than LP or QP problems

Subsequent lectures – Dynamic programming

- Leads to a ***globally optimal*** solution for very general discrete-time optimal control problems
- Can be very computationally intensive, but still more efficient than an exhaustive grid search