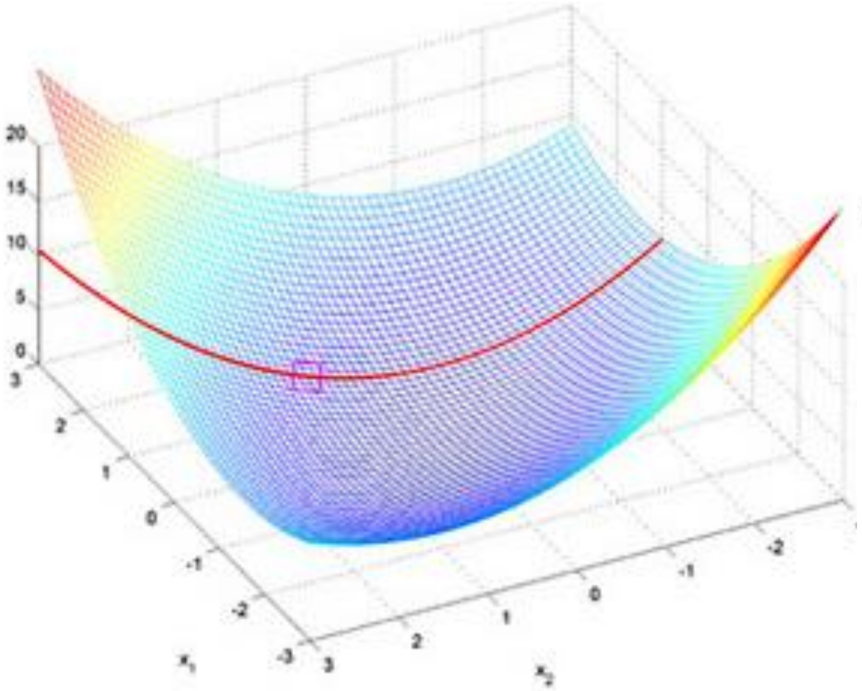
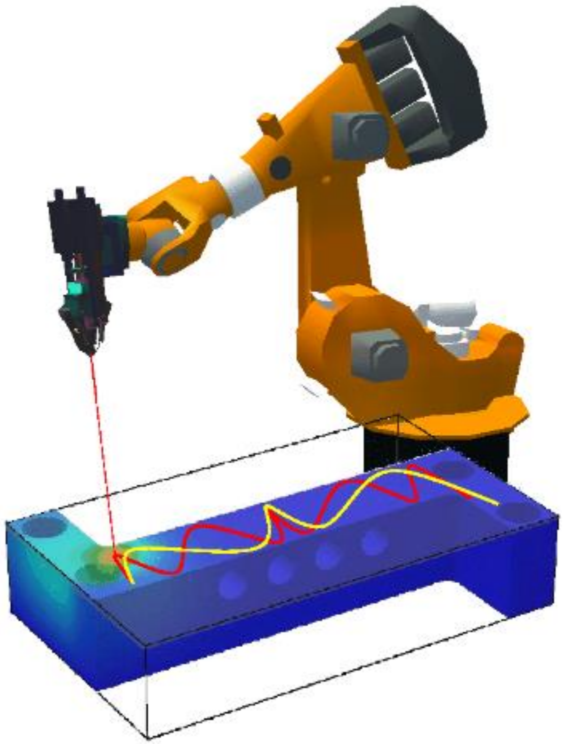
MEGR 3090/7090/8090: Advanced Optimal Control



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

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

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



Lecture 10

September 21, 2017

Karush-Kuhn-Tucker (KKT) Conditions - Reminder

Optimization problem: Minimize $J(\mathbf{u})$

Subject to: $g(\mathbf{u}) \leq \mathbf{0}$
$h(\mathbf{u}) = \mathbf{0}$

Optimality requirements: $\nabla J(\mathbf{u}^*) + \boldsymbol{\mu}^T \nabla g(\mathbf{u}^*) + \boldsymbol{\lambda}^T \nabla h(\mathbf{u}^*) = 0$

Complementary slackness: $\mu_i g_i(\mathbf{u}^*) = 0, \forall i$

Combines additional terms from inequality and equality constraints

Feasibility: $\mu_i \geq 0, \forall i$

Constraint satisfaction: $g(\mathbf{u}^*) \leq 0$
$h(\mathbf{u}^*) = \mathbf{0}$

Note: If $\mathbf{u}^*$ is a minimizer, then the KKT conditions will be satisfied (i.e., they are necessary). In general, however, satisfaction of the KKT conditions does not guarantee that u^* is a unique global minimizer (or even a minimizer)!

Karush-Kuhn-Tucker (KKT) Conditions - Challenges

Challenges w/ KKT conditions:

1) Only sufficient for global optimality if the problem is globally convex.
2) Equations are always nonlinear w/ inequality constraints, due to complementary slackness.
   - Often nonlinear in other cases.

Classic Convex Optimization Problems - Reminder

Linear programming (LP) optimization problem:

Minimize: $J(\mathbf{u}) = \boldsymbol{k}^T \mathbf{u}$

Subject to:
$$\begin{aligned} A_1 \mathbf{u} - \mathbf{b}_1 &\le \mathbf{0} \\ A_2 \mathbf{u} - \mathbf{b}_2 &= \mathbf{0} \end{aligned}$$

Quadratic programming (QP) optimization problem:

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

Subject to:
$$\begin{aligned} A_1 \mathbf{u} - \mathbf{b}_1 &\le \mathbf{0} \\ A_2 \mathbf{u} - \mathbf{b}_2 &= \mathbf{0} \end{aligned}$$

Constrained Linear Quadratic Regulator (LQR) - Reminder

Optimal control problem:

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

Subject to: $$M_1 u(i) - b_1 \leq 0, i = 0 \ldots N-1$$
$$M_2 \mathbf{x}(i) - \mathbf{b}_2 \leq \mathbf{0}, i = 0 \ldots N-1$$
$$\mathbf{x}(i+1) = A\mathbf{x}(i) + Bu(i)$$

Reminder: Last lecture, we proved this optimization problem was convex.

Exercise: Prove that this optimization problem reduces to a QP once the recursion relating J entirely to $\mathbf{u}$ and $\mathbf{x}_0$ has been performed.

Constrained Linear Quadratic Regulator (LQR) - Reminder

Constrained Linear quadratic regulator: $J(\underline{u};\underline{x}(0)) = \sum_{i=0}^{N-1} [\underline{x}(i+1)^T Q \underline{x}(i+1) + R u(i)^2]$

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

$$M_1 u(i) - b_1 \leq 0, \quad i = 0 \ldots N-1$$

$$M_2 \underline{x}(i) - \underline{b}_2 \leq 0, \quad i = 0 \ldots N-1$$

$$\Rightarrow M_2 [h_i \underline{u} + A^i \underline{x}(0)] - \underline{b}_2 \leq 0, \quad i = 1 \ldots N$$

$$\Rightarrow M_2 h_i \underline{u} \underbrace{+ M_2 A^i \underline{x}(0) - \underline{b}_2}_{\text{constant vector}} \leq 0, \quad i = 1 \ldots N$$

$\Rightarrow$ Fits the QP formulation.

KKT Conditions for Linear Programs – Assessment

Optimal control problem:

Minimize: $J(\mathbf{u}) = \boldsymbol{k}^T\mathbf{u}$ ← 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

LP-specific KKT conditions:

$\boldsymbol{k}^T + \boldsymbol{\mu}^T A_1 + \boldsymbol{\lambda}^T A_2 = \mathbf{0}$ ← p **linear** equations

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

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

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

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

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

Key observations:

- 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

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: $2\mathbf{u}^T Q + \mathbf{r}^T + \boldsymbol{\mu}^T A_1 + \boldsymbol{\lambda}^T A_2 = \mathbf{0}$ ← 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$^{\text{th}}$ row of A_1, b_{1i} = i$^{\text{th}}$ 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

Complementary slackness: $\mu_i (A_{1,i} \underline{u}^* - \underline{b}_{1,i}) = 0, \ i = 1 \ldots q$

Either: $\mu_i \neq 0 \Rightarrow \boxed{A_{1,i} \underline{u}^* - \underline{b}_{1,i} = 0} \leftarrow$ Linear

$A_{1,i} \underline{u}^* - \underline{b}_{1,i} \neq 0 \Rightarrow \boxed{\mu_i = 0} \leftarrow$ Linear

Active Set Methods for Convex Optimization - Introduction

"Problem equation" for LP and QP: $\mu_i(A_{1i}\mathbf{u}^* - b_{1i}) = 0, i = 1 \dots q$

Key observations:

- 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

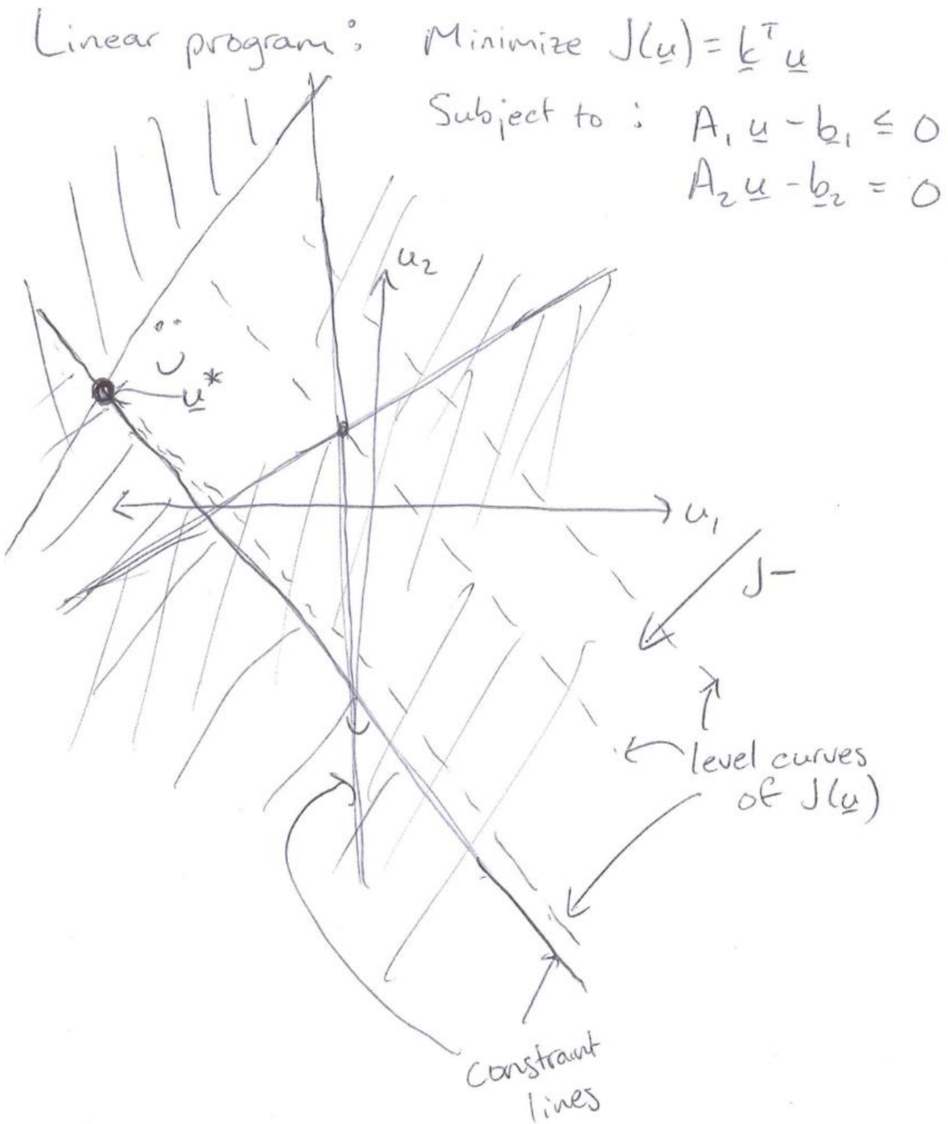
Key idea of active set methods:

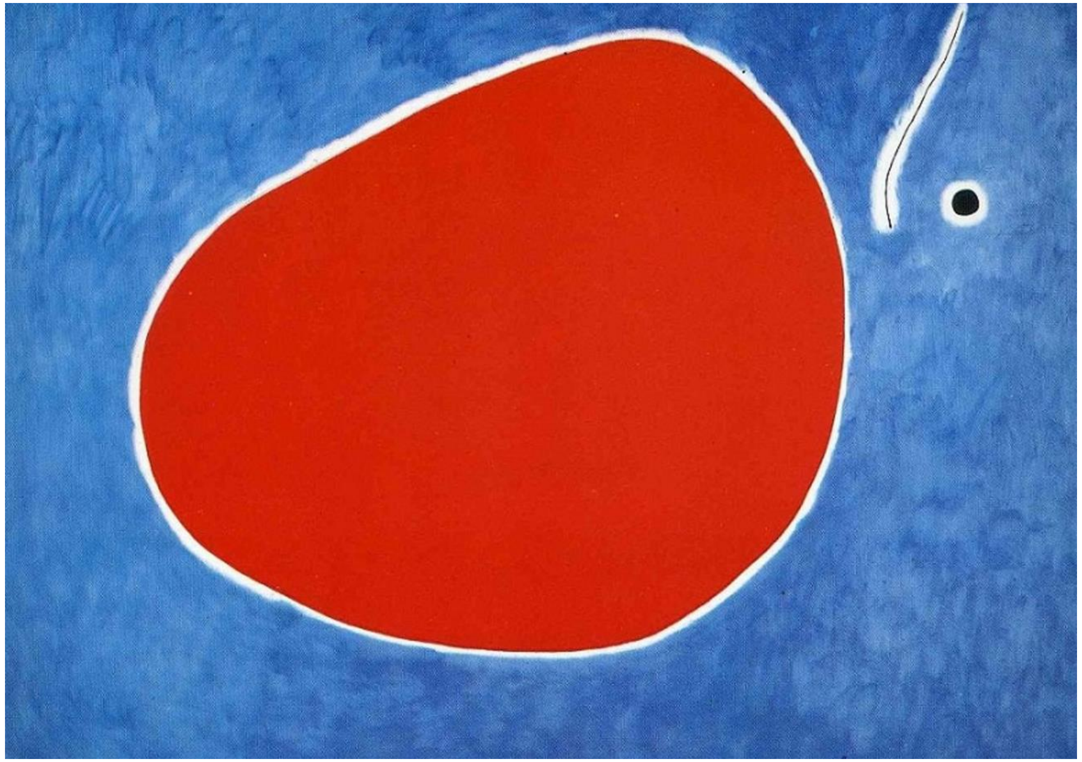
- Assume that certain constraints are active (i.e.,$A_{1i}\mathbf{u}^* - b_{1i}, \mu_i \neq 0$) and the others are inactive (i.e., $\mu_j = 0, A_{1j}\mathbf{u}^* - b_{1j} \neq 0$)
- Solve the KKT conditions under the above assumptions
- Try other combinations of active vs. inactive constraints to see which one leads to a solution that satisfies $\mu_i \geq 0, i = 1 \dots q$ and $\lambda_i \neq 0, i = 1 \dots r$

Simplex Method – An LP-Specific Active Set Numerical Solution Approach

The works of two very talented artists are shown at the right.

Let me know if you want a framed version of the picture on the left. I'm sure we can agree on a fair price.





Simplex Method – An LP-Specific Active Set Numerical Solution Approach

Notation:

$$G_1 \triangleq \{\mathbf{u}: A_1\mathbf{u} - \mathbf{b}_1 \leq \mathbf{0}\}$$
$$G_2 \triangleq \{\mathbf{u}: A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}\}$$

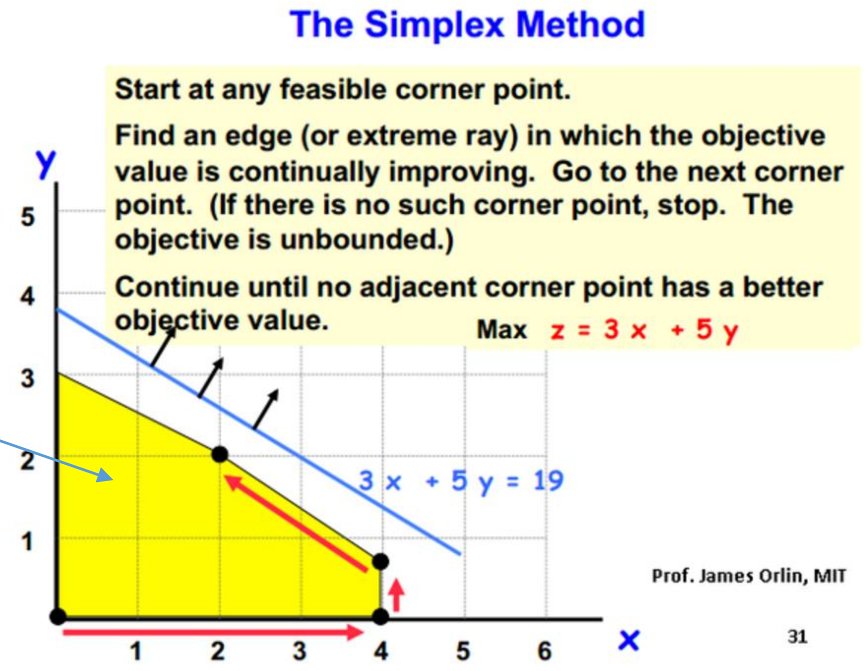
G_1 and G_2 are the inequality and equality constraint **sets**, respectively.

Key fact: Suppose that, in addition to being **convex** (which we already proved), $G_1 \cap G_2$ is also **closed and bounded**. Then the unique global minimizer, $\mathbf{u}^*$, **lies at a vertex** of $G_1 \cap G_2$.

Conceptual idea of the simplex method (illustrated below):

- Start at a vertex of $G_1 \cap G_2$
- Move along an edge, in the direction of decreasing J, until another vertex is reached
- Stop when a vertex is reached where it is not possible to move in the direction of decreasing J (this is the unique global minimizer, $\mathbf{u}^*$)
- Note: Only at the unique global minimizer will we have $\mu_i \geq 0, i = 1 \ldots q$ and $\lambda_i \neq 0, i = 1 \ldots r$

$G_1 \cap G_2$



Simplex Method – An LP-Specific Active Set Numerical Solution Approach

UNC CHARLOTTE

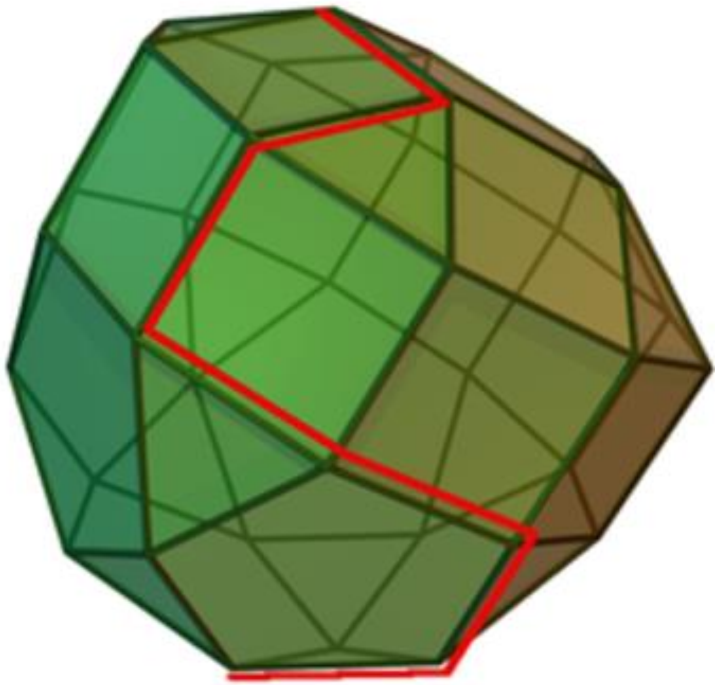
Notation: $G_1 \triangleq \{\mathbf{u}: A_1\mathbf{u} - \mathbf{b}_1 \leq \mathbf{0}\}$
$G_2 \triangleq \{\mathbf{u}: A_2\mathbf{u} - \mathbf{b}_2 = \mathbf{0}\}$

G_1 and G_2 are the inequality and equality constraint **sets**, respectively.

Key fact: Suppose that, in addition to being **convex** (which we already proved), $G_1 \cap G_2$ is also **closed and bounded**. Then the unique global minimizer, $\mathbf{u}^*$, **lies at a vertex** of $G_1 \cap G_2$.

In higher dimensions, a closed and bounded set $\boldsymbol{G_1 \cap G_2}$ is a **polytope** (see figure at right)

The simplex algorithm moves from vertex to vertex of the polytope, in the direction of decreasing J. Because of the convexity of $G_1 \cap G_2$ and linearity of J, this procedure will always lead to the **global** minimizer.



Simplex Method – Built in MATLAB Function for Linear Programming

UNC CHARLOTTE

Syntax:

```
options = optimoptions('linprog','Algorithm','dual-simplex');
```

```
[u_opt,J_opt] = linprog(k,A1,b1,A2,b2,[],[],options);
```

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

Documentation available at: https://www.mathworks.com/help/optim/ug/linprog.html

Simplex Method for Linear Programming - Example

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

Subject to:

$$u_1 + 2u_2 - 8 \leq 0$$

$$u_1 - u_2 - 1.5 \leq 0$$

$$-2u_1 + 1 \leq 0$$

$$-2u_2 + 1 \leq 0$$

Note: MATLAB code is posted on Canvas

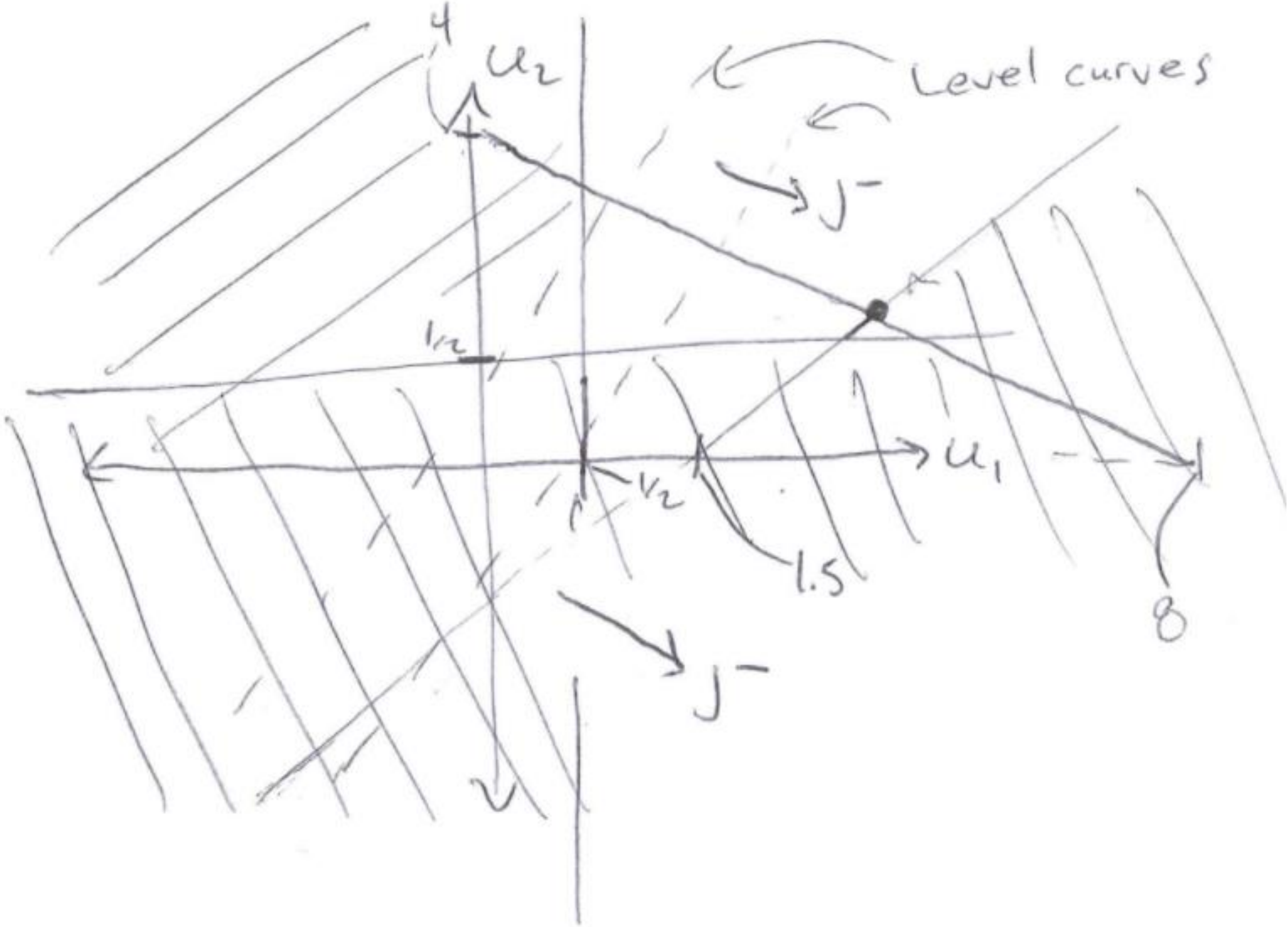
Simplex Method for Linear Programming - Example

Example: $J(\underline{u}) = -2u_1 + u_2 = \overbrace{\begin{bmatrix} -2 & 1 \end{bmatrix}}^{\underline{c}^T} \underline{u}$

Constraints:
$$\left.\begin{aligned} u_1 + 2u_2 - 8 &\leq 0 \\ u_1 - u_2 - 1.5 &\leq 0 \\ -2u_1 + 1 &\leq 0 \\ -2u_2 + 1 &\leq 0 \end{aligned}\right\} \begin{bmatrix} 1 & 2 \\ 1 & -1 \\ -2 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} - \begin{bmatrix} 8 \\ 1.5 \\ -1 \\ -1 \end{bmatrix} \leq \underline{0}$$

Simplex Method for Linear Programming - Example

Just when you thought the artwork couldn't get any better…



General Procedure for Active Set Methods (from Papalambros)

Source:
Papalambros page 303

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.

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 $\mathbf{u}^*$

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

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.

Preview of Upcoming Lectures

Next lecture – Interior point methods for obtaining numerical solutions to convex optimization problems

- Barrier functions
- Penalty functions
- Use of interior point methods for quadratic programming (QP)

Subsequent lectures – Dealing with more general (possibly non-convex) optimization problems:

- Sequential quadratic programming