Distributed Real-Time Control Systems

Module 17

Distributed Optimization

Coordination by Optimization

A control task objective can be defined by the minimization of a cost function:

$$\min_{\mathbf{u}} J(\mathbf{x}, \mathbf{u})$$

$$s.t.$$

$$\mathbf{x} \in \mathcal{X}$$

$$\mathbf{u} \in \mathcal{U}$$

$\mathbf{x}$ – vector of state variables

$\mathbf{u}$ – vector of control variables

$\mathcal{X}$ – constraints on state variables

$\mathcal{U}$ – constraints on control variables

Optimization in distributed systems

The global cost function depends on the local state $\boldsymbol{x_i}$ and control variables $\boldsymbol{u_i}$, for each subsystem $\boldsymbol{i}$. The optimization problem can be written as:

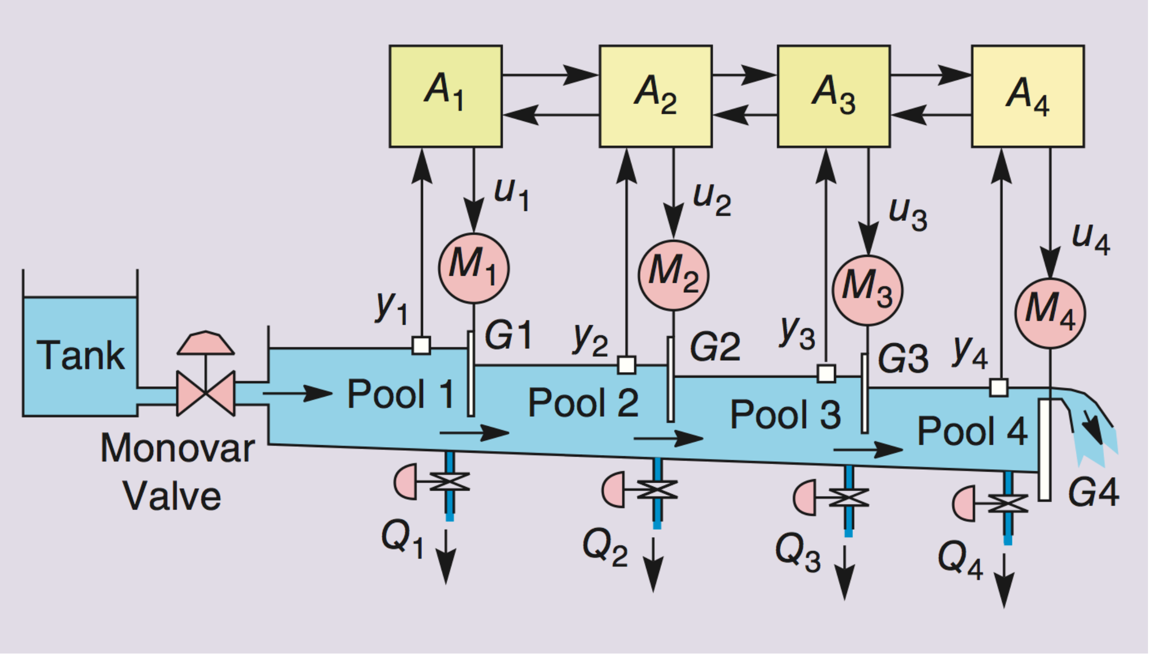
$$(\mathbf{u_1^*}, \cdots, \mathbf{u_n^*}) = \underset{\mathbf{u_1}, \cdots, \mathbf{u_n}}{\operatorname{argmin}} \; J(\mathbf{x_1}, \cdots, \mathbf{x_n}, \mathbf{u_1}, \cdots, \mathbf{u_n})$$

$$s.t.$$

$$(\mathbf{x_1}, \cdots, \mathbf{x_n}) \in \mathcal{X}$$

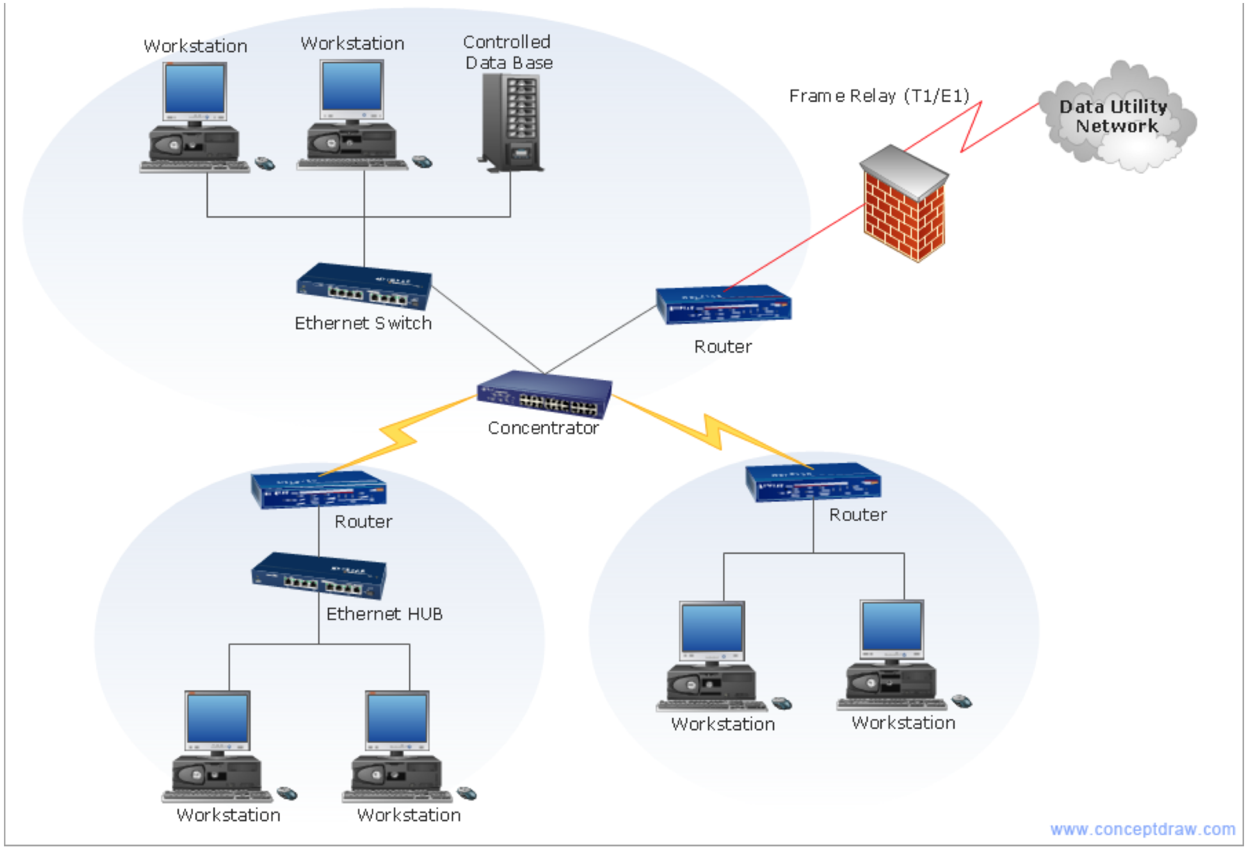
$$(\mathbf{u_1}, \cdots, \mathbf{u_n}) \in \mathcal{U}$$

Example: Water Distribution



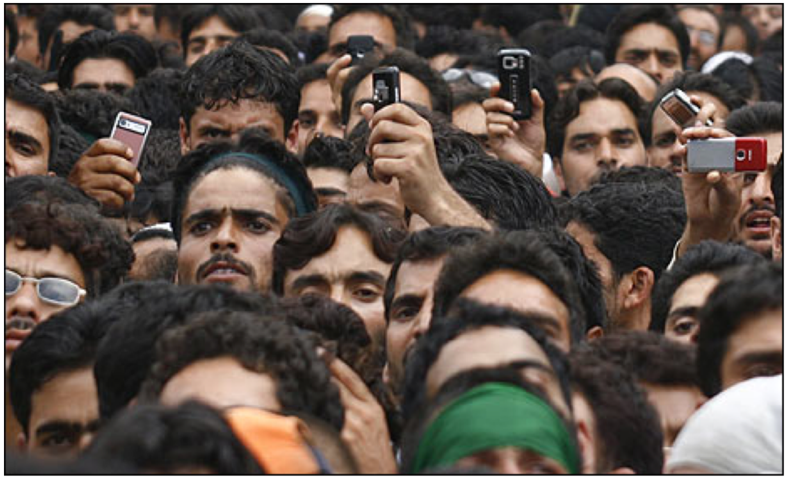
$$J_i = \frac{1}{2}\sum_{k=1}^{\infty}\left[e_i^2(k) + \rho_i u_i^2(k)\right]$$

Example: Internet Congestion Control



$$\begin{array}{ll} \text{maximize} & \sum_v u_v(s_v) \\ \text{subject to} & Rs \preceq c \end{array}$$

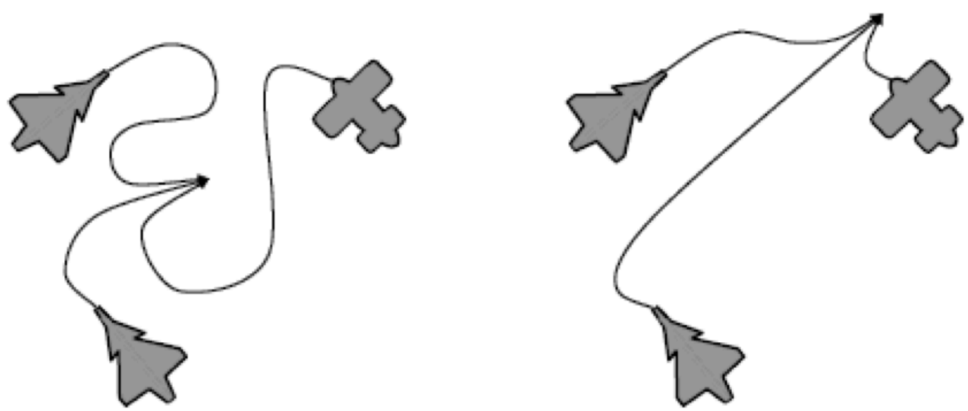
Example: Estimation in Sensor Networks



$$\min \frac{1}{2} \sum_i \left(x_i - \theta \right)^2$$

Example: Vehicle Coordination

$$
\begin{aligned}
f_v(\theta) \quad &= \min_u \; \textstyle\sum_{t=0}^{T} (x_t - \theta)^T Q (x_t - \theta) + u_t^T R u_t \\
&\text{s.t.} \quad x_{t+1} = A x_t + B u_t,\ t = 0, \ldots, T-1
\end{aligned}
$$



Rendez-vous problem

Solving Optimization Problems

What problems do we know how to solve with mathematical optimization?

Least squares:

Closed form solutions

$$\theta' = \operatorname{argmin} \frac{1}{2} \sum_i \left(\theta^T x_i - y_i\right)^2$$

Linear Programs:

Simplex like algorithms

$$\begin{aligned} &\text{minimize } c^T x \\ &\text{subject to } Ax = b \end{aligned}$$

Quadratic Programs

E.g. optimal Control

$$\begin{aligned} &\text{minimize } x^T Q x + c^T x \\ &\text{subject to } Ax = b \end{aligned}$$

Convex Optimization:

Duality theory, (sub)gradient methods.

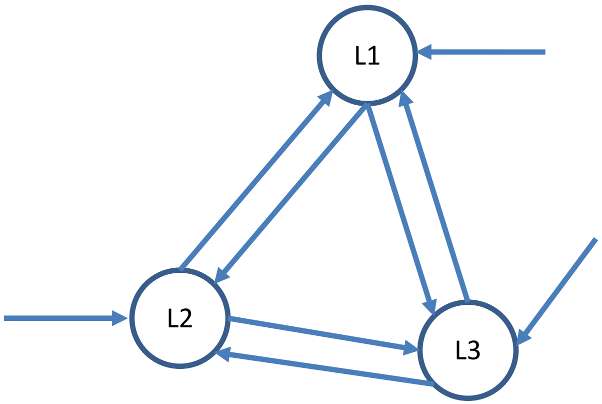
$$\begin{aligned} &\text{minimize } f_0(x) \\ &\text{subject to } x \in X \end{aligned}$$

A few non convex problems:

E.g. SVD

Illumination System as a Constrained Optimization Problem

- i – index of the desk/node
- l_i – illuminance at desk i
- L_i – lower bound of illuminance at desk i
- d_i – dimming level of each led (percentage).
- K_{ij} – Influence of the luminance of led j in the irradiance at desk i.
- o_i – effect of external illumination at desk i.
- n – number of nodes



$$\min_{d_1,\cdots,d_n} f(d_1,\cdots,d_n) \qquad \text{some criteria to optimize}$$

$$s.t.$$

$$l_i = \sum_{j=1}^{n} d_j K_{ij} + o_i \geq L_i,\ \forall i \qquad \text{illuminance above desired level}$$

$$0 \leq d_i \leq 100,\ \forall i \qquad \text{the constraints of the actuator}$$

Calibration

- *When is it necessary to recalibrate K_{ij} ?*

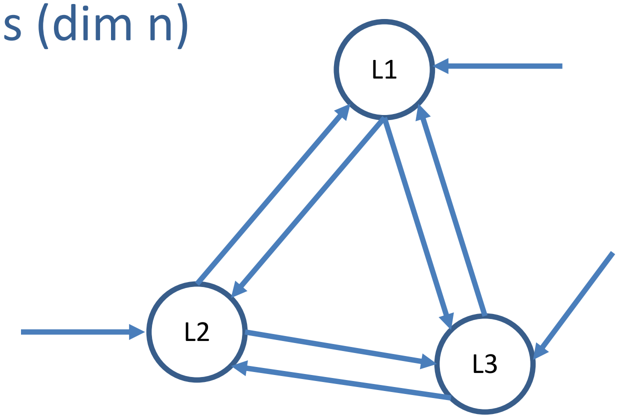
- Luminaires change position/orientation
- Desks change position/orientation
- Other elements in the environment change position/orientation -> reflected light will be different.

External Illuminance

- *How to estimate the external illuminance o_i in run time?*
- If the external illumination is constant (or slowly varying) a possible algorithm is:
  - Make a prediction of the expected illuminance at each desk assuming zero external illuminance.
  - Subtract the actual measured illuminance.
  - Do some low pass filtering to reduce sensor noise.

Vector Notation

- **l** – columns vector with all illuminances (dim n)
- **d** – column vector with all normalized dimming (dim n)
- **K** – Coupling matrix (dim n by n)
- **o** – columns vector of external illuminances (dim n)
- **L** – column vector of lower bound illuminances (dim n)



$$\min_{\mathbf{d}} f(\mathbf{d})$$

$$s.t.$$

$$\mathbf{Kd} + \mathbf{o} \geq \mathbf{L}$$

$$\mathbf{0} \leq \mathbf{d} \leq \mathbf{100}$$

0 – column vector with all elements = 0
100 – column vector with all elements = 100

The constraint set

$$K\mathbf{d} + \mathbf{o} \geq \mathbf{L}$$

$$\mathbf{0} \leq \mathbf{d} \leq \mathbf{100}$$

- Convex set of R^n
- Non empty if $\mathbf{L} \leq K\mathbf{100} + \mathbf{o}$
- If the requested minimum desired illuminance is greater than what is possible with all leds at full power plus the external illuminance, the problem is **unfeasible**.
  - *What to do in this case ?*

Example: 2D constraint set

$$K = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$K\mathbf{d} + \mathbf{o} \geq \mathbf{L}$$
$$\mathbf{0} \leq \mathbf{d} \leq \mathbf{100}$$

CONSTRAINT 1 (C1): $2d_1 + d_2 + o_1 \geq L_1 \implies d_2 \geq L_1 - o_1 - 2d_1$

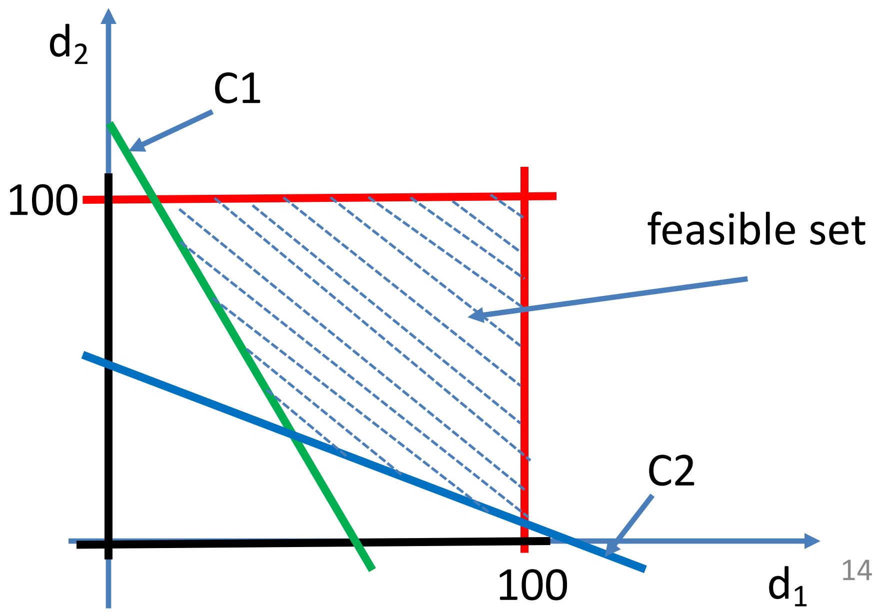
CONSTRAINT 2 (C2): $d_1 + 2d_2 + o_2 \geq L_2 \implies d_2 \geq \dfrac{L_2 - o_2}{2} - \dfrac{1}{2}d_1$

CONSTRAINT 3 (C3): $d_1 \geq 0$

CONSTRAINT 4 (C4): $d_1 \leq 100$

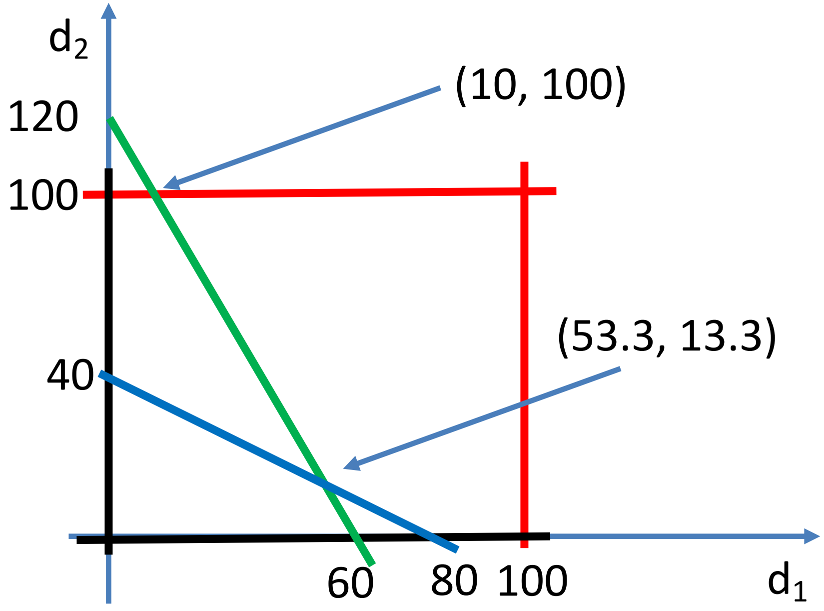
CONSTRAINT 5 (C5): $d_2 \geq 0$

CONSTRAINT 6 (C6): $d_2 \leq 100$

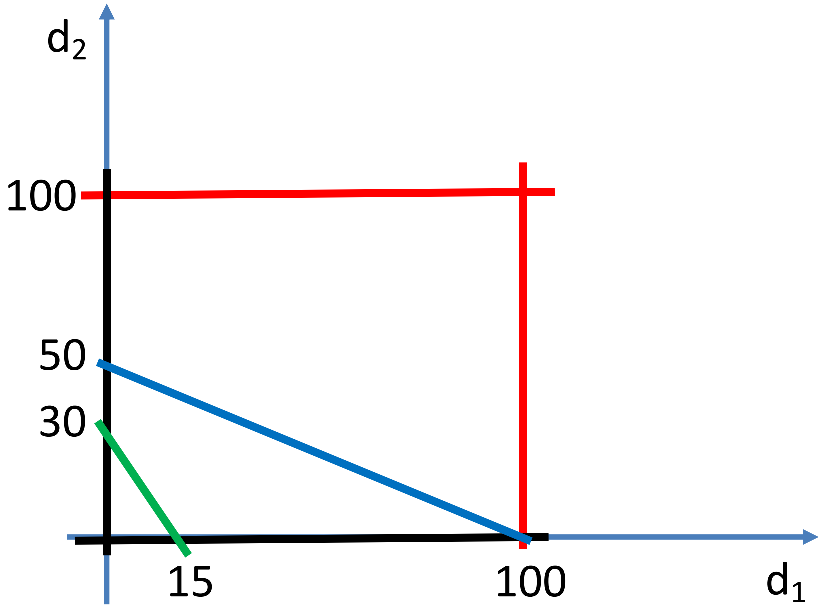


Example: 2D constraint set

Case 1:
L1 = 150 , L2 = 80
o1 = 30 , o2 = 0



Case 2:
L1 = 80 , L2 = 150
o1 = 50 , o2 = 50



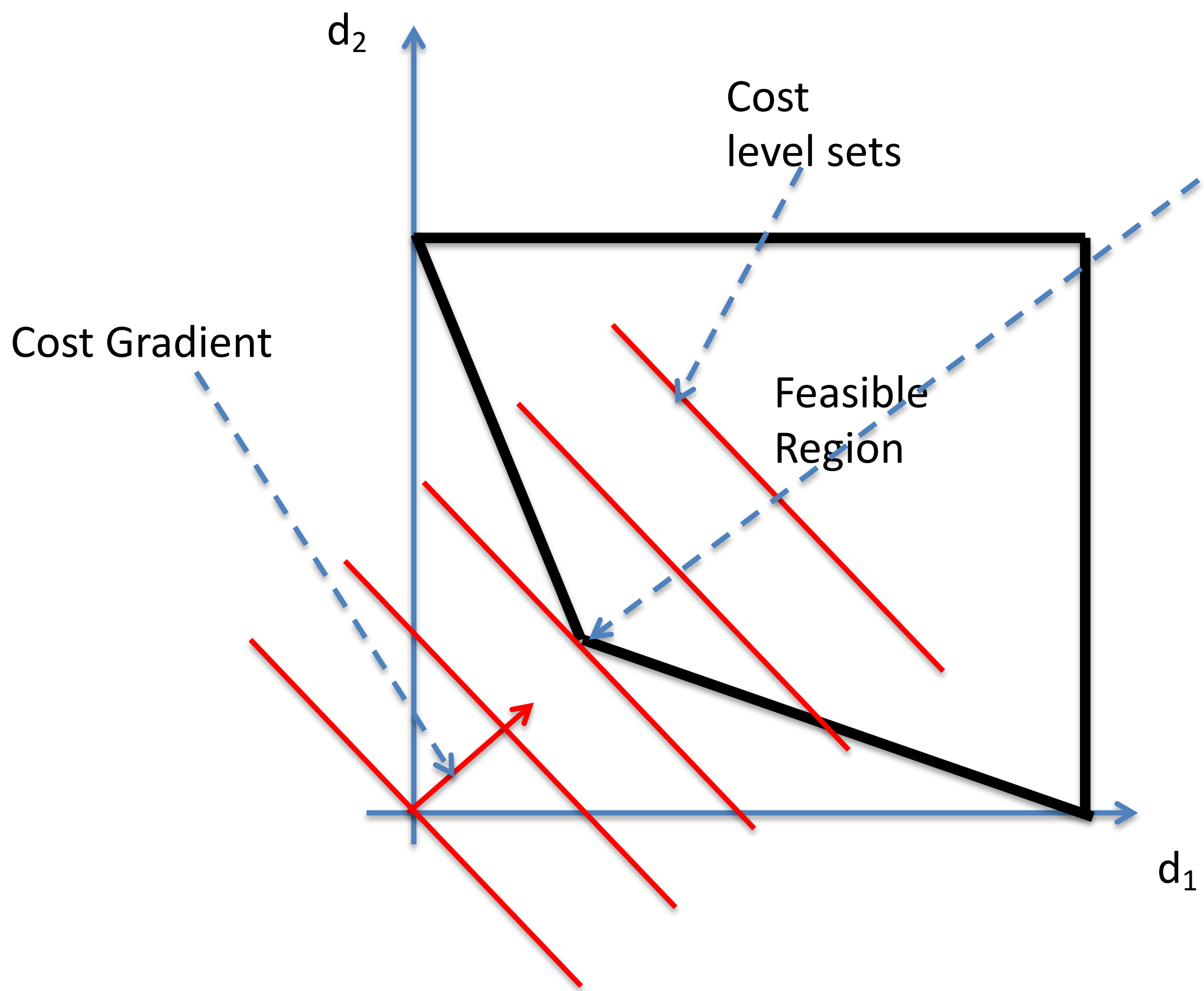
Cost function: Minimize Energy

$$f(\mathbf{d}) = c_1 d_1 + \cdots + c_n d_n = \mathbf{c}^T \mathbf{d}$$

- c_i – the "cost" of energy at desk *i*.
- **c** – the vector of all energy costs.

- => LINEAR PROGRAM

Solution of Linear Program



The optimal solutions are found where the cost level lines (orthogonal to the gradient) first meet the feasible region.

It will always be at the boundary of the feasible region. One of the "corners" of the feasible set will be a solution.

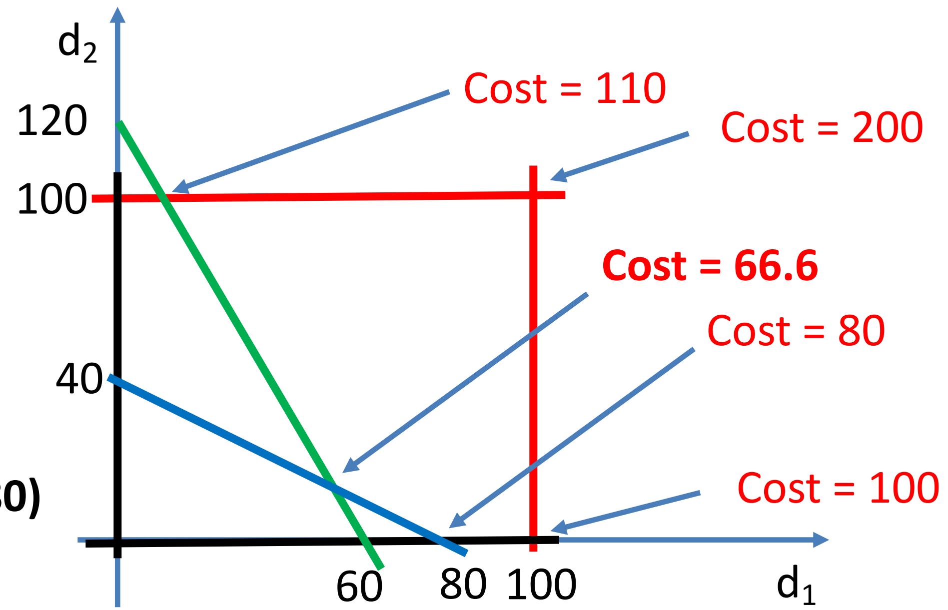
Example: Identical Costs

c1 = 1 , c2 = 1

Case 1:
L1 = 150 , L2 = 80
o1 = 30 , o2 = 0

<u>Optimal Solution:</u>

(d1, d2) = (53.3, 13.3) => (l1 , l2) = (150, 80)



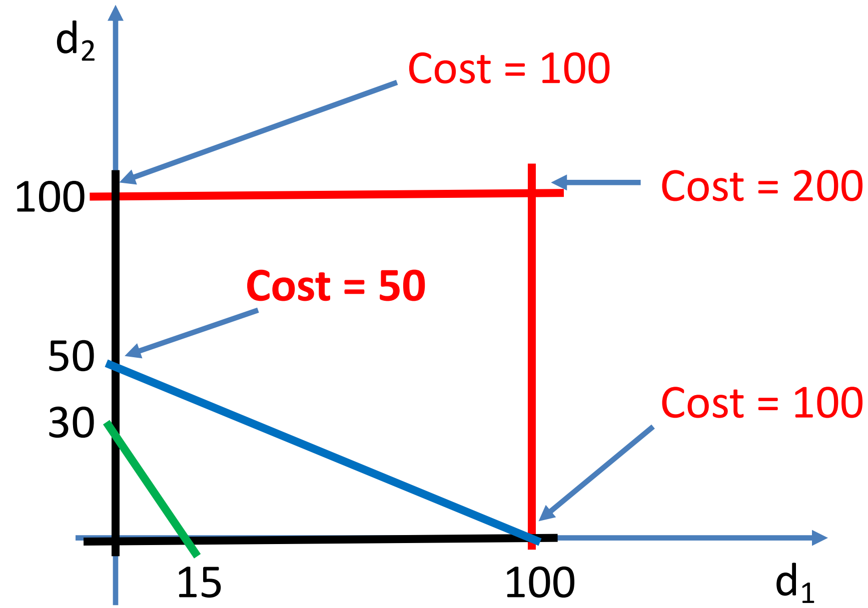
Note: In this case the optimal solution results in exactly in the minimum desired illuminance in both desks.

Example: Identical Costs

c1 = 1 , c2 = 1

Case 2:
L1 = 80 , L2 = 150
o1 = 50 , o2 = 50

Optimal Solution:
(d1, d2) = (0, 50) => (l1 , l2) = (100, 150)

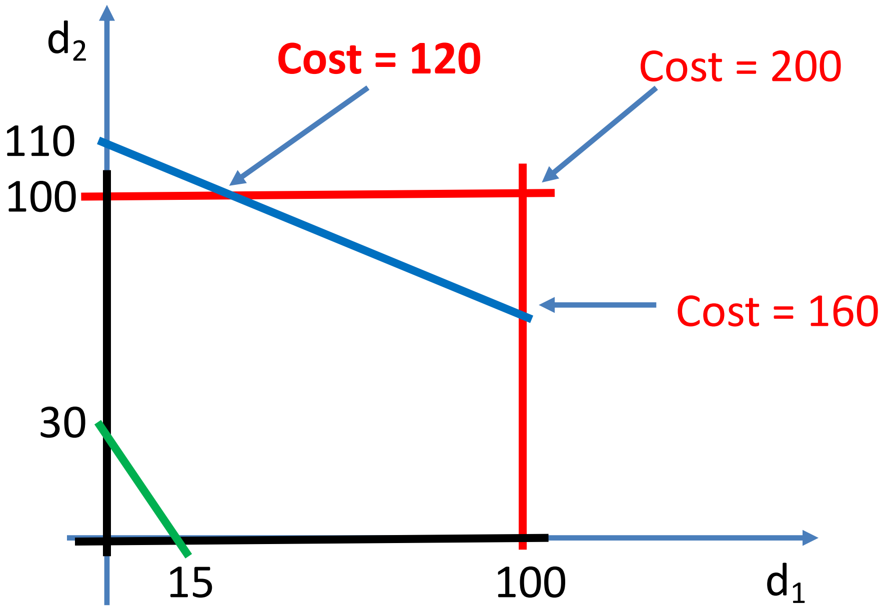


Note: In this case the optimal solution results in the minimum desired illuminance in desk 2, but the luminance in **desk 1 exceeds** the minimum. Why ?

Example: Identical Costs

$c1 = 1 \;,\; c2 = 1$

Case 3:
L1 = 80 , L2 = 270
o1 = 50 , o2 = 50



Optimal Solution:
(d1, d2) = (20, 100) => (l1 , l2) = (190, 270)

Note: In this case the optimal solution results in the minimum desired illuminance in desk 2, but the luminance in **desk 1 exceeds** the minimum. Why ?

Questions

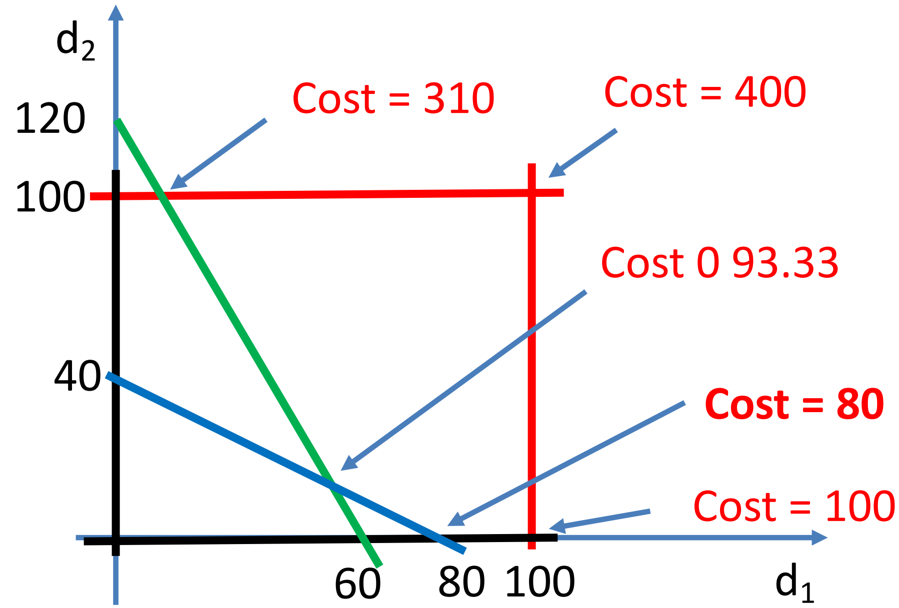
- Assume the dynamics of the coupled system is stable. What would be the solution of the problem with the uncoordinated strategy (independent local controllers) ? Consider all cases.

- Same thing, for the coordinated solution by Feedforward from Accessible Disturbance.

Example: Asymmetrical Costs

$c1 = 1 \;,\; c2 = 3$

Case 1:
L1 = 150 , L2 = 80
o1 = 30 , o2 = 0



Optimal Solution:
(d1, d2) = (80, 0) => (l1 , l2) = (190, 80)

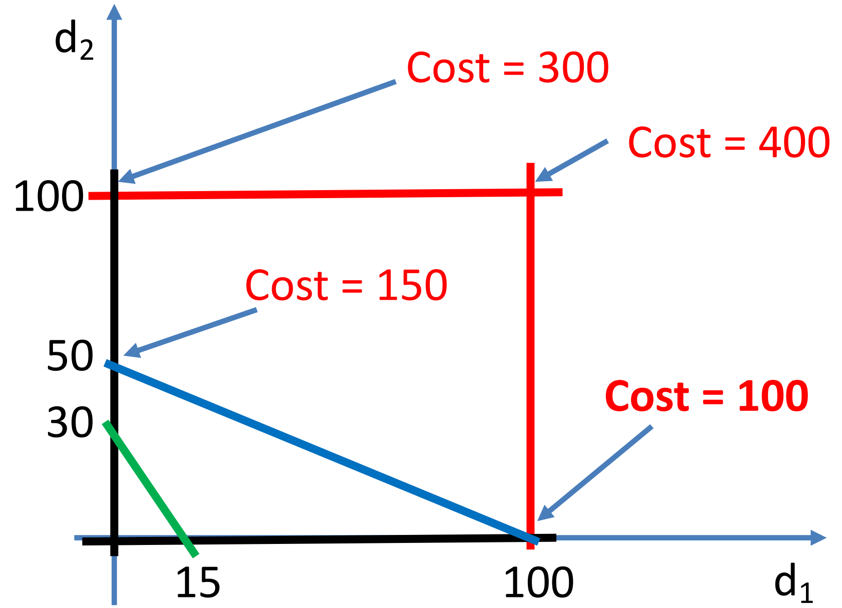
Note: In this case the optimal solution results in the minimum desired illuminance in desk 2, but the luminance in **desk 1 exceeds** the minimum. Why ?

Example: Asymmetrical Costs

$c1 = 1 \ , \ c2 = 3$

Case 2:
L1 = 80 , L2 = 150
o1 = 50 , o2 = 50

Optimal Solution:
(d1, d2) = (100, 0) => (l1 , l2) = (210, 150)



Note: In this case the optimal solution results in the minimum desired illuminance in desk 2, but the luminance in **desk 1 exceeds** the minimum. Why ?

Example: Identical Costs

$c1 = 1 \;,\; c2 = 3$

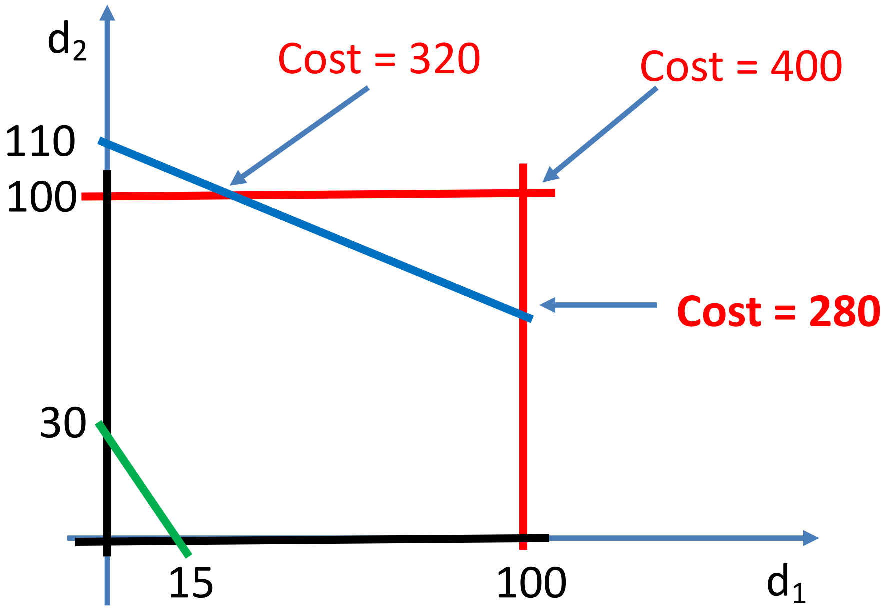
Case 3:

L1 = 80 , L2 = 270

o1 = 50 , o2 = 50

Optimal Solution:

(d1, d2) = (100, 60) => (l1 , l2) = (310, 270)



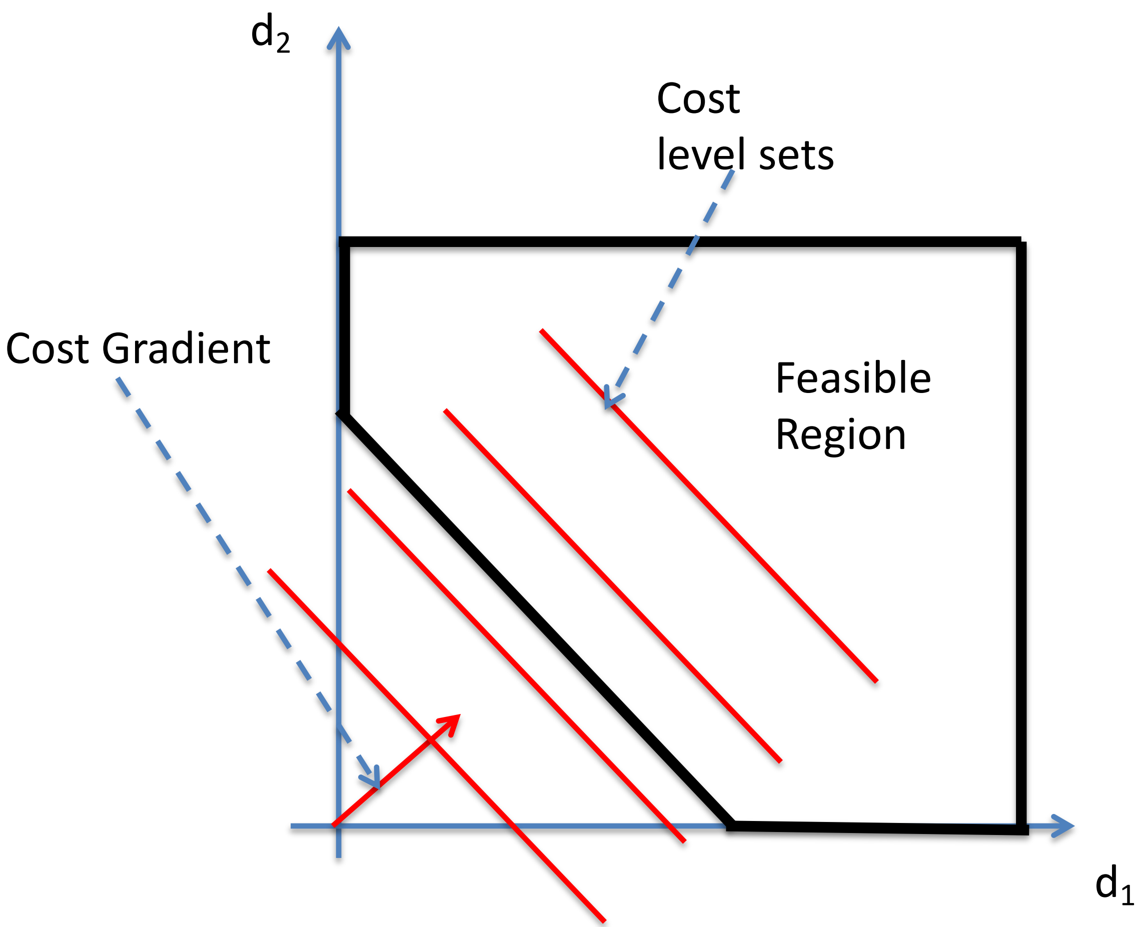
Note: In this case the optimal solution results in the minimum desired illuminance in desk 2, but the luminance in **desk 1 exceeds** the minimum. Why ?

Important Note

- The solutions from optimization result in “altruistic” solutions.
- A luminaire may actively decide to exceed its minimum required value to “help” other luminaires.
- Thus, a simple local rule to set the target at the reference values is not optimizing our cost function.
- New references to the feedforward and feedback controllers must then be set in order to implement the solution of the optimization problem.

Problems with Linear Programs

- Solutions may be degenerate



What is the optimal solutions in this case ?

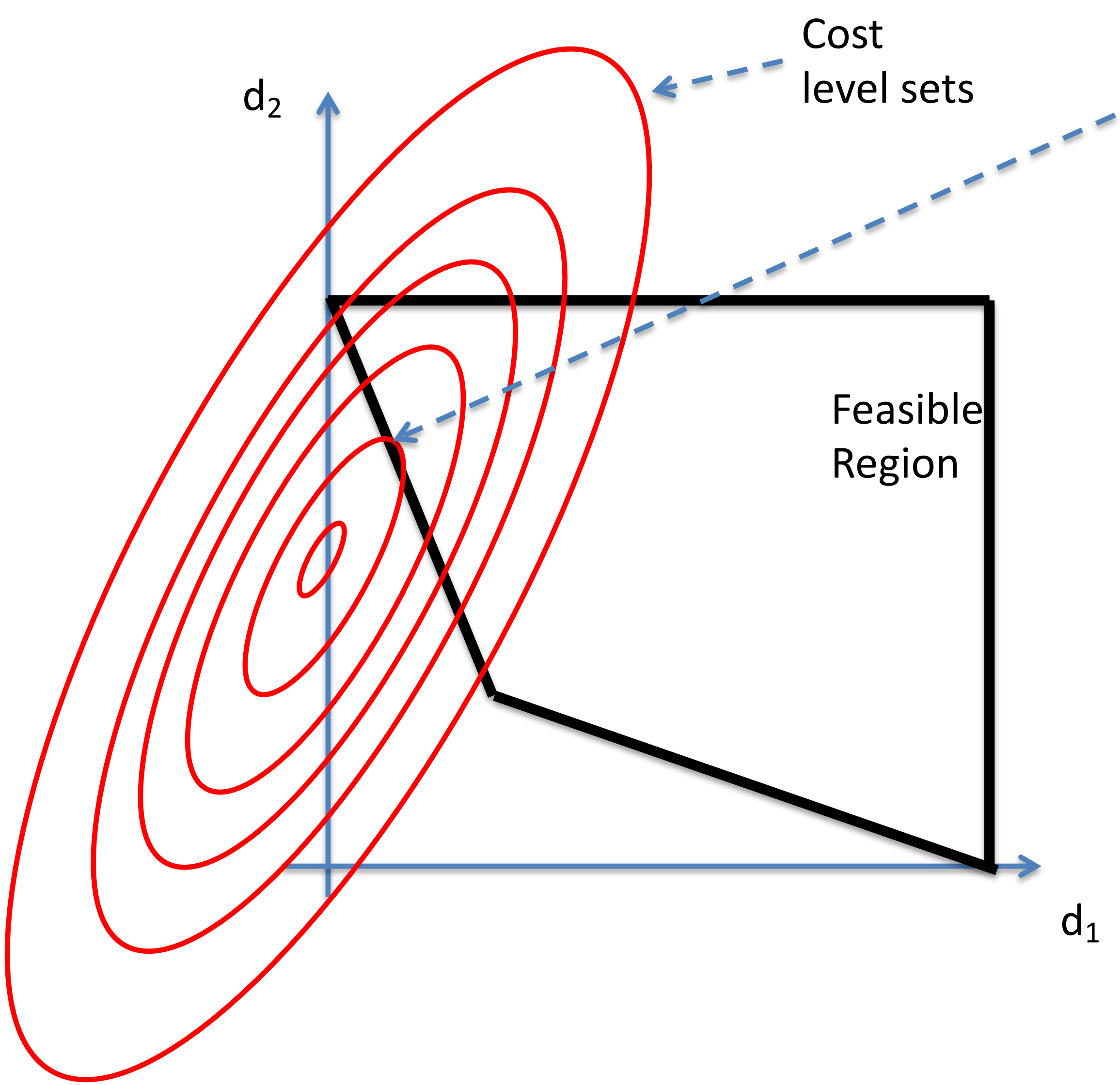
Degenerate solutions tend to be troublesome for solvers.

Cost function: Minimize Energy and Maximize Lamp Longevity

$$f(\mathbf{d}) = \mathbf{c}^T\mathbf{d} + \mathbf{d}^T\mathbf{Q}\mathbf{d}$$

- Solution will prevent lamps going maximum power, which will increase lamp longevity.
- **Q** = diag(q_1, …, q_n) – cost of the lamps at time unit per max power.
- => QUADRATIC PROGRAM

Solution of Quadratic Program



The optimal solutions is at the optimum of the quadratic function, or where the cost level curves first meet the feasible region.

It will be either at :
1 - the optimum of the quadratic function or
2- at corner points or
3 - at points where the gradients of the cost and constraints have the same direction

Example: 2D constraint set

$$K = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$K\mathbf{d} + \mathbf{o} \geq \mathbf{L}$$

$$\mathbf{0} \leq \mathbf{d} \leq \mathbf{100}$$

CONSTRAINT 1 (C1): $2d_1 + d_2 + o_1 \geq L_1 \implies d_2 \geq L_1 - o_1 - 2d_1$

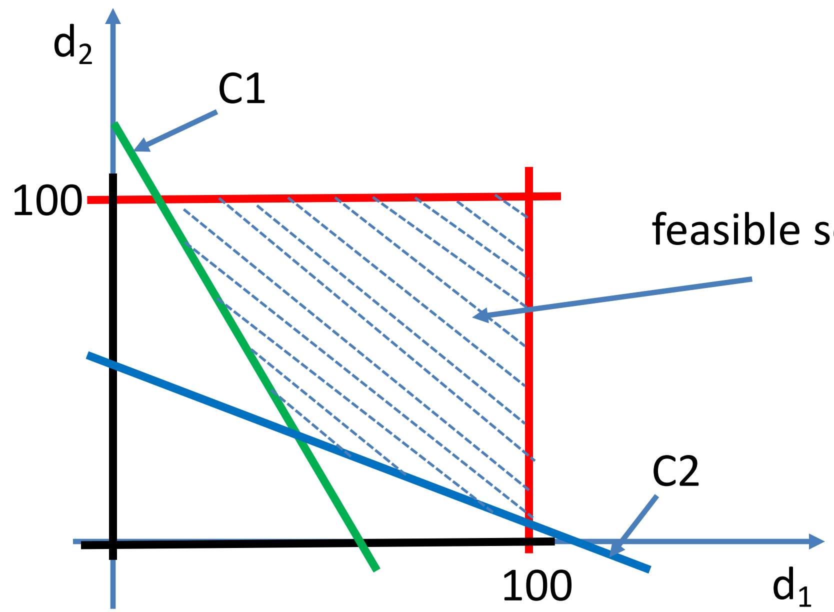
CONSTRAINT 2 (C2): $d_1 + 2d_2 + o_2 \geq L_2 \implies d_2 \geq \frac{L_2 - o_2}{2} - \frac{1}{2}d_1$

CONSTRAINT 3 (C3): $d_1 \geq 0$

CONSTRAINT 4 (C4): $d_1 \leq 100$

CONSTRAINT 5 (C5): $d_2 \geq 0$

CONSTRAINT 6 (C6): $d_2 \leq 100$



Solution by Search

- For the case of the previous slide a brute force solutions would be to compute value of f(d) at the following points:
  1. argmin of f(d), if feasible
  2. intersection of the boundaries of C1 with C2, if feasible
  3. intersection of the boundaries of C1 with C3, if feasible
  4. intersection of the boundaries of C1 with C4, if feasible
  5. intersection of the boundaries of C1 with C5, if feasible
  6. intersection of the boundaries of C1 with C6, if feasible
  7. intersection of the boundaries of C2 with C3, if feasible
  8. intersection of the boundaries of C2 with C4, if feasible
  9. intersection of the boundaries of C2 with C5, if feasible
  10. intersection of the boundaries of C2 with C6, if feasible
  11. intersection of the boundaries of C3 with C5, if feasible
  12. intersection of the boundaries of C4 with C6, if feasible
  13. Gradient of f = gradient of C1, if feasible
  14. Gradient of f = gradient of C2, if feasible
  15. Gradient of f = gradient of C3, if feasible
  16. Gradient of f = gradient of C4, if feasible
  17. Gradient of f = gradient of C5, if feasible
  18. Gradient of f = gradient of C6, if feasible

Fortunately, there are much more efficient solutions!