MATLAB for Optimization

Zhanle (Gerald) Wang

University of Regina

Outline

- Introduction of mathematical optimization
- How to use optimization to model and solve engineering problems
- How to solve optimization models using MATLAB and CVX
- Applications
  - Production plan
  - Smart electric vehicle charging
  - Vehicle-to-grid
  - Cast some machine learning to optimization
  - Stochastic optimization
- Solving optimization problems

Outline

- Introduction of mathematical optimization
- How to use optimization to model and solve engineering problems
- How to solve optimization models using MATLAB and CVX
- Applications
  - Production plan
  - Smart electric vehicle charging
  - Vehicle-to-grid
  - Cast some machine learning to optimization
  - Stochastic optimization
- Solving optimization problems
- The slides and all the codes are available at my website: http://uregina.ca/∼wang233z

What is an optimization model?

- A mathematical approach for seeking a “best” “decision/action” from a “set of alternatives”
- An objective function that is to be maximized or minimized
- A set of constraints (possibly empty) that must be satisfied

Formal Formulation

- Mathematical programs are problems of the form

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \le 0,\ \forall i = 1, ..., m \end{aligned} \tag{1}$$

  - $x \in \mathbb{R}^n$ is the optimization variable
  - $f : \mathbb{R}^n \to \mathbb{R}$ is objective function
  - $g_i : \mathbb{R}^n \to \mathbb{R}$ are (inequality) constraint functions
- Feasible region: $\mathcal{C} = x : g_i(x) \le 0,\ \forall i = 1, ..., m$
- $x^\star \in \mathbb{R}^n$ is an optimal solution if $x^\star \in \mathcal{C}$, and $f(x^\star) \le f(x),\ \forall x \in \mathcal{C}$

Minimization vs Maximization

- Without loss of generality, it is sufficient to consider a minimization objective since
  $\underset{x}{\text{maximize}}\ \{f(x) : x \in \mathcal{C}\} \equiv -\underset{x}{\text{minimize}}\ \{f(x) : x \in \mathcal{C}\}$
- Thus to develop the theory we will only consider minimization problems. When actually solving problems we can use the actual min or max objective

Program vs Optimization Problem

- A “program” or “mathematical program” is an optimization problem with a finite number of variables and constraints written out using explicit mathematical (algebraic) expressions
- The word “program”, “programming” means “plan”, “planning”
- Early applications of optimization arose in planning resource allocations and gave rise to “programming” to mean optimization (predates computer programming)
- We will use “program”, “programming” and “optimization problem”, “optimization” interchangeably

Problem Classification

- The tractability of a large scale optimization problem depends on the structure of the functions that make up the objective and constraints, and the domain restrictions on the variables.

Functions	Variables	Problem type	Difficulty

Problem Classification

- The tractability of a large scale optimization problem depends on the structure of the functions that make up the objective and constraints, and the domain restrictions on the variables.

Functions	**Variables**	**Problem type**	**Difficulty**
All linear	Continuous variables	Linear Program (LP) or Linear Optimization problem	easy

Problem Classification

- The tractability of a large scale optimization problem depends on the structure of the functions that make up the objective and constraints, and the domain restrictions on the variables.

Functions	**Variables**	**Problem type**	**Difficulty**
All linear	Continuous variables	Linear Program (LP) or Linear Optimization problem	easy
Some nonlinear	Continuous variables	Nonlinear Program (NLP)or Nonlinear Optimization Problem	Easy or Difficult

Problem Classification

- The tractability of a large scale optimization problem depends on the structure of the functions that make up the objective and constraints, and the domain restrictions on the variables.

Functions	**Variables**	**Problem type**	**Difficulty**
All linear	Continuous variables	Linear Program (LP) or Linear Optimization problem	easy
Some nonlinear	Continuous variables	Nonlinear Program (NLP)or Nonlinear Optimization Problem	Easy or Difficult
Linear/nonlinear	Some discrete	Integer Program (IP) or Discrete optimization problem	Difficult

Develop optimization models

- Many problems of real importance can be formulated as an optimization problem
- There are two important aspects
  - Mathematical modeling of engineering problems
  - Problem solving
- Reducing a seemingly new problem to an instance of a well-known problem allows one to use pre-existing methods for solving them

Formulation Steps

- Encode decisions/actions as *decision variables* whose values we are seeking
- Identify the relevant *problem data*
- Express *constraints* on the values of the decision variables as mathematical relationships (inequalities) between the variables and problem data
- Express the *objective function* as a function of the decision variables and the problem data

Ex1: Production plan

- A company produces two types of drinks: Drink 1 and Drink 2 and they yield different profit.
- The capacity of the barrels to contain Drink 1 and Drink 2 are 500 L and 400 L.
- Each type drink requires some amount of ingredient A and B.
- There are 30 L ingredient A and 44 L ingredient B in stock.
- The manager is developing a plan to maximize the profit.

Drink (L)	**A (L)**	**B (L)**	**Profit ($)**
Drink 1	0.03	0.08	1.00
Drink 2	0.06	0.04	1.25

Formulation

- *Decision variables*:

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*:

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:
  - Ingredient constraints:

$$0.03x_1 + 0.08x_2 \leq 30$$

$$0.08x_1 + 0.04x_2 \leq 44$$

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:
  - Ingredient constraints:

$$0.03x_1 + 0.08x_2 \leq 30$$

$$0.08x_1 + 0.04x_2 \leq 44$$

  - Barrel constraints: $x_1 \leq 500, x_2 \leq 400$

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:
  - Ingredient constraints:

$$0.03x_1 + 0.08x_2 \leq 30$$

$$0.08x_1 + 0.04x_2 \leq 44$$

  - Barrel constraints: $x_1 \leq 500, x_2 \leq 400$
  - Production constraints:

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:
  - Ingredient constraints:

$$0.03x_1 + 0.08x_2 \leq 30$$

$$0.08x_1 + 0.04x_2 \leq 44$$

  - Barrel constraints: $x_1 \leq 500, x_2 \leq 400$
  - Production constraints: $x_1 \geq 0, x_2 \geq 0$

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:
  - Ingredient constraints:

$$0.03x_1 + 0.08x_2 \leq 30$$

$$0.08x_1 + 0.04x_2 \leq 44$$

  - Barrel constraints: $x_1 \leq 500, x_2 \leq 400$
  - Production constraints: $x_1 \geq 0, x_2 \geq 0$
- *Objective function*:

Formulation

- *Decision variables*: amount of drinks
  i.e., x_1: Drink 1; x_2: Drink 2
- *Data*: required ingredient to produce drinks, amount of ingredient in stock, barrel capacity, profit of each drink
- *Constraints*:
  - Ingredient constraints:
    $$0.03x_1 + 0.08x_2 \leq 30$$
    $$0.08x_1 + 0.04x_2 \leq 44$$
  - Barrel constraints: $x_1 \leq 500, x_2 \leq 400$
  - Production constraints: $x_1 \geq 0, x_2 \geq 0$
- *Objective function*: $1x_1 + 1.25x_2$

Formulation

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & -(1x_1 + 1.25x_2) \\
\text{subject to} \quad & 0.03x_1 + 0.06x_2 \leq 30 \\
& 0.08x_1 + 0.04x_2 \leq 44 \\
& x_1 \leq 500 \\
& x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{aligned}
\tag{2}
$$

Formulation

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & -(1x_1 + 1.25x_2) \\
\text{subject to} \quad & 0.03x_1 + 0.06x_2 \leq 30 \\
& 0.08x_1 + 0.04x_2 \leq 44 \\
& x_1 \leq 500 \\
& x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{aligned}
\tag{2}
$$

- Linear programming

Formulation

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & -(1x_1 + 1.25x_2) \\
\text{subject to} \quad & 0.03x_1 + 0.06x_2 \leq 30 \\
& 0.08x_1 + 0.04x_2 \leq 44 \\
& x_1 \leq 500 \\
& x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{aligned} \tag{2}
$$

- Linear programming
- can be solve by $linprog$ using MATLAB

MATLAB's Optimization Toolbox

- Not a complete list

Problem type	**Solvers**
Linear Programming	$linprog$
Mixed Linear Programming	$intlinprog$
Linear Least Squares	$lsqlin, lsqnonneg$
Quadratic Programming	$quadprog$

Linear programming solver

- $linprog(c, A, b, Aeq, beq, lb, ub, options)$
- Finds the minimum of a problem specified by

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & c^T x \\ \text{subject to} \quad & Ax \leq b \\ & A_{eq}x = b_{eq} \\ & lb \leq x \leq ub \end{aligned} \tag{3}$$

where c, x, b, beq, lb, and ub are vectors, and A and Aeq are matrices.

Model reformulation

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & 1x_1 + 1.25x_2 \\
\text{subject to} \quad & 0.03x_1 + 0.06x_2 \leq 30 \\
& 0.08x_1 + 0.04x_2 \leq 44 \\
& x_1 \leq 500 \\
& x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & c^T x \\
\text{subject to} \quad & Ax \leq b \\
& A_{eq}x = b_{eq} \\
& lb \leq x \leq ub
\end{aligned}
$$

Model reformulation

- What are A, b, c, x?

Model reformulation

- What are A, b, c, x?

$$A = \begin{bmatrix} 0.03 & 0.06 \\ 0.08 & 0.04 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \quad b = \begin{bmatrix} 30 \\ 44 \\ 500 \\ 400 \\ 0 \\ 0 \end{bmatrix} \quad c = \begin{bmatrix} 1 \\ 1.25 \end{bmatrix}$$

Model reformulation

- What are A, b, c, x?

$$A = \begin{bmatrix} 0.03 & 0.06 \\ 0.08 & 0.04 \\ 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \quad b = \begin{bmatrix} 30 \\ 44 \\ 500 \\ 400 \\ 0 \\ 0 \end{bmatrix} \quad c = \begin{bmatrix} 1 \\ 1.25 \end{bmatrix}$$

- Matlab code: $x = linprog(c, A, b)$
  here:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Model reformulation (alternative)

- $x = linprog(c, A, b, [\,], [\,], lb, ub)$

$$A = \begin{bmatrix} 0.03 & 0.06 \\ 0.08 & 0.04 \end{bmatrix} \; b = \begin{bmatrix} 30 \\ 44 \end{bmatrix} \; c = \begin{bmatrix} 1 \\ 1.25 \end{bmatrix}$$

$$lb = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \; ub = \begin{bmatrix} 500 \\ 400 \end{bmatrix} \; x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Quadratic programming solver

- $quadprog(H, c, A, b, Aeq, beq, lb, ub)$
- Finds the minimum of a problem specified by

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & \frac{1}{2}x^T Hx + c^T x \\
\text{subject to} \quad & Ax \leq b \\
& A_{eq}x = b_{eq} \\
& lb \leq x \leq ub
\end{aligned}
\tag{4}
$$

  where c, x, b, beq, lb, and ub are vectors, and H, A and Aeq are matrices.
- It is harder to reformulate a problem to standard quadratic program.

CVX

- CVX is a Matlab-based modeling system for convex optimization.
- CVX turns Matlab into a modeling language, allowing constraints and objectives to be specified using standard Matlab expression syntax.

CVX code

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & 1x_1 + 1.25x_2 \\
\text{subject to} \quad & 0.03x_1 + 0.06x_2 \leq 30 \\
& 0.08x_1 + 0.04x_2 \leq 44 \\
& x_1 \leq 500 \\
& x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

CVX code

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & 1x_1 + 1.25x_2 \\
\text{subject to} \quad & 0.03x_1 + 0.06x_2 \leq 30 \\
& 0.08x_1 + 0.04x_2 \leq 44 \\
& x_1 \leq 500 \\
& x_2 \leq 400 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

```
cvx_begin
  variable x(2);
  minimize -(1*x(1)+
    1.25*x(2))
  subject to
  0.03*x(1)+0.06*x(2)<=30;
  0.08*x(1)+0.04*x(2)<=44;
  x(1)<=500;
  x(2)<=400;
  x(1)>=0;
  x(2)>=0;
cvx_end
```

CVX Python code

```
import numpy as np
import cvxpy as cp
n = 2
x = cp.Variable(n)
objective = cp.Minimize(-(100*x[0]+125*x[1]))
constraints = [3*x[0]+6*x[1]<=30,
               8*x[0]+4*x[1]<=44,
               x[0]<=5,
               x[1]<=4,
               x[0]>=0,
               x[1]>=0]
prob = cp.Problem(objective, constraints)
result = prob.solve()
print(x.value)
```

Electric vehicles (EV)

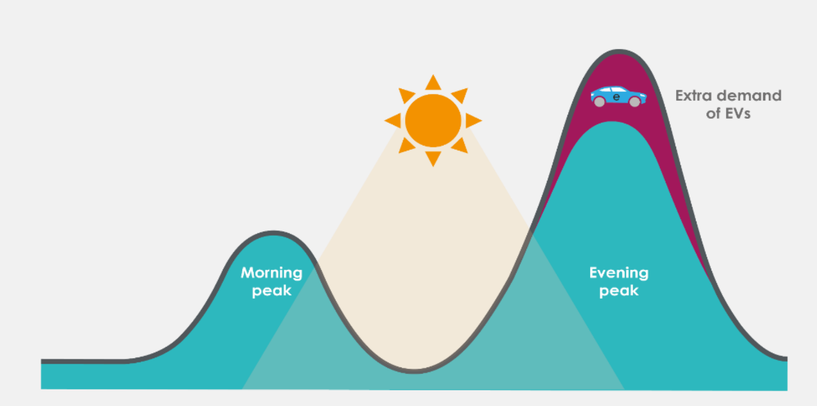
- Lower maintenance costs
- Lower taxes
- Cheaper fuel (electricity)
- Government subsidy

Ref: fleetcarma.com

Basic EV charging

- Typical battery capacity: $30kWh - 100kWh$
- Onboard charger: 1.9 – 22 kW
- DC offboard charger: 50 – 350 kW
- Charging level
  - Level 1: 0 – 10 kW
  - Level 2: 10 – 50 kW
  - Level 3: 50 – 350 kW

Impact on power system



Ref: Electric Cars, edx.org

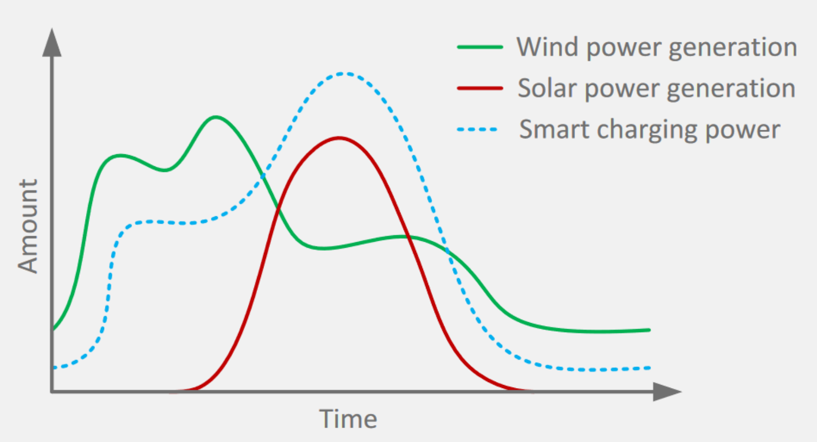
Smart charging and vehicle-to-grid (V2G)

- Using the electric vehicle battery to feed power back to the grid using a bidirectional EV charger
- Advantages: storage for renewables, reduce peak demand, ancillary services, etc
- Challenges: optimal control, bidirectional charger, battery degradation, standardization, regulatory framework, etc

V2G applications

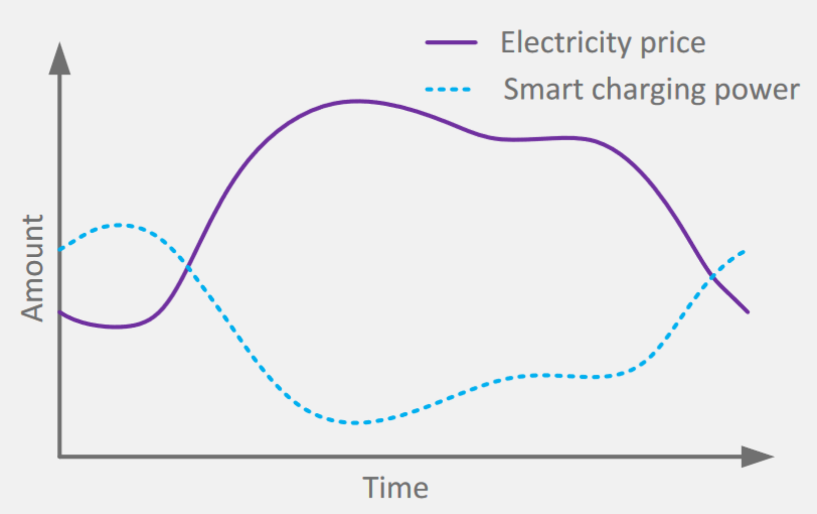
- Local load balancing
  - Adjust charging time/power according to load
  - Balance multiple charge points with priority
- Renewable energy utilization
- Price based charging/discharging
- Peak shaving
- Grid back up

Renewable energy availability



Ref: Electric Cars, edx.org

Price based charging

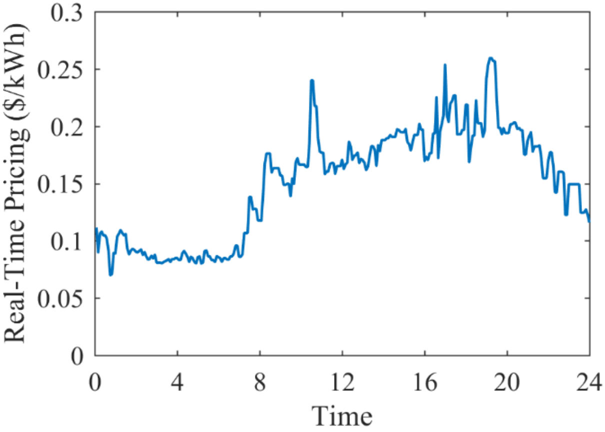


Ref: Electric Cars, edx.org

Ex2: Smart charging

- You have an EV and can be charged at home.
- The electricity price is real-time pricing (RTP).
- You need $20kW$ to charge up your EV.
- The rated power of the charger is $2kW$.
- Assume that the EV can be charged anytime in the day.
- Now, you need to determine what time to charge to minimize the electricity bill in the day.

Real-time-pricing



Ref: Calculated from locational marginal price on October 27, 2019 in PJM market

Smart charging model

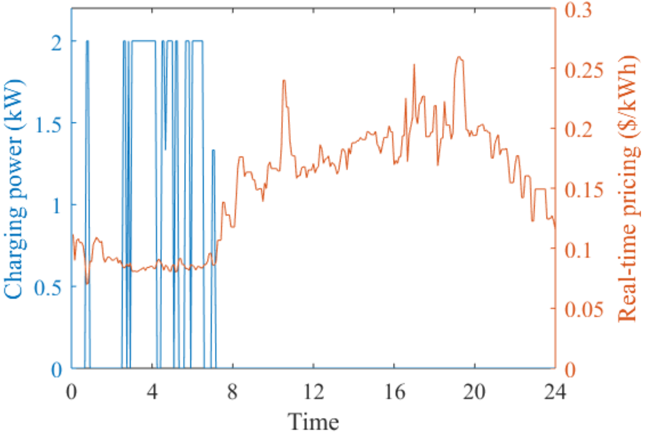
$$\begin{aligned} \underset{t}{\text{minimize}} \quad & \sum_{t \in \mathcal{T}} \pi_t p_t \\ \text{subject to} \quad & \sum_{t \in \mathcal{T}} p_t = 20 \\ & 0 \leq p_t \leq 2 \end{aligned}$$

where π_t is price, p_t is charging power.

CVX code

```
load('As3_data')
price = price';
cvx_begin
    variable p(288)
    minimize price * p
    subject to
        sum(p)==20*60/15; %kW15min
        0<=p<=2;
cvx_end
```

Smart charging



Smart charging?

- Is this too aggressive?

Smart charging?

- Is this too aggressive?
- How about battery degradation cost?

Smart charging?

- Is this too aggressive?
- How about battery degradation cost?
- How about
  - home arriving, leaving and driving time?

Smart charging?

- Is this too aggressive?
- How about battery degradation cost?
- How about
  - home arriving, leaving and driving time?
  - state of charge?

Smart charging?

- Is this too aggressive?
- How about battery degradation cost?
- How about
  - home arriving, leaving and driving time?
  - state of charge?
  - multiple vehicles?
  - convenience and privacy?
  - power system reliability?

Ex3: V2G

$$\underset{P_t^{EV_i}}{\text{minimize}}\ \operatorname{var}\left(P_t^{EV_i} - P_{t\in[t_1,t_2]\cup[t_3,t_4]}^{EV_i} + P_t^{EV_{-i}} + P_{base,t}\right)$$

$$+\,\upsilon \sum_{t\in\mathcal{T}} DC(DOD_t)\left|P_t^{EV_i}\right| + \kappa \sum_{t\in\mathcal{T}} RTP_t\ P_t^{EV_i}$$

subject to:

$$-P_{rated} \leq P_t^{EV_i} \leq P_{rated}, \forall\, t \in [t_2, t_3] \cup [t_4, t_1]$$

$$\sum_t P_t^{EV_i} = \beta E_0, \ \forall\, t \in [t_1, t_2] \cup [t_3, t_4]$$

$$SOC_t = SOC_0 + \sum_{t=0}^{t} P_t^{EV_i}$$

$$SOC_{min} \leq SOC_t \leq SOC_{max}, \forall\, t \in \mathcal{T}$$

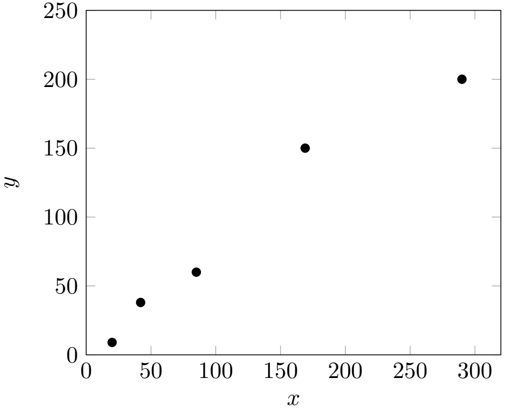
$$SOC_{t_2} \geq SOC_{acc}$$

$$SOC_{t_4} \geq SOC_{acc}$$

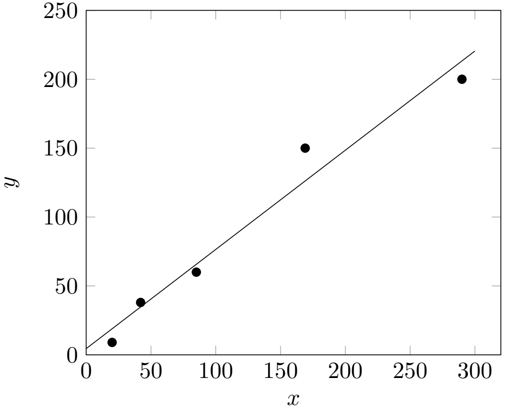
$$DOD_t = 1 - SOC_t, \quad \forall\, t \in \mathcal{T}$$

Ref: K. Ginigeme and Z. Wang, "Distributed Optimal Vehicle-To-Grid Approaches with Consideration of Battery Degradation Cost under Real-Time Pricing", IEEE Access, 2020.

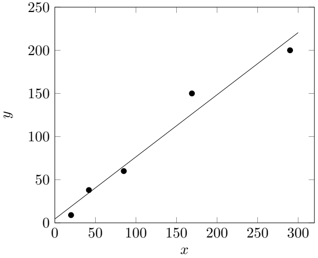
Machine learning: data fitting



Machine learning: data fitting



Machine learning: data fitting



- Given $x_i, y_i, i = 1, ..., m$, find $f(\theta) = \theta_1 * x + \theta_2$ that optimizes

$$\underset{\theta}{\text{minimize}} \sum_{i=1}^{m} (\theta_1 x_i + \theta_2 - y_i)^2$$

where θ_1 is slope, θ_2 is intercept.

Formal problem setting

- Input: $x_i \in \mathbb{R}^n, i = 1, ..., m$
- Output: $y_i \in \mathbb{R}, i = 1, ..., m$
- Model parameters: $\theta \in \mathbb{R}^k$
- Predicted output: $\hat{y} \in \mathbb{R}$

Formal problem setting

- Input: $x_i \in \mathbb{R}^n, i = 1, ..., m$
- Output: $y_i \in \mathbb{R}, i = 1, ..., m$
- Model parameters: $\theta \in \mathbb{R}^k$
- Predicted output: $\hat{y} \in \mathbb{R}$
- Let's define a function that maps inputs to feature vectors

$$\phi : \mathbb{R}^n \to \mathbb{R}^k$$

- Then, we can write

$$\hat{y}_i = \sum_{j=1}^{k} \theta_j \cdot \phi_j(x_i) \equiv \theta^T \phi(x_i)$$

Formal problem setting

$$\hat{y}_i = \sum_{i=1}^{j} \theta_j \cdot \phi_j(x_i) \equiv \theta^T \phi(x_i)$$

- Let's write it in a more compact way

$$\Phi \in \mathbb{R}^{m \times k} = \begin{bmatrix} \phi(x_1)^T \\ \phi(x_2)^T \\ \vdots \\ \phi(x_m)^T \end{bmatrix}, \; y \in \mathbb{R}^m = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

Then,

$$\hat{y} = \Phi\theta$$

Loss function

- The loss function is defined as

$$\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$$

- Square loss

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

- Absolute loss

$$\ell(\hat{y}, y) = |\hat{y} - y|$$

- Deadband loss

$$\ell(\hat{y}, y) = max\{0, |\hat{y} - y| - \varepsilon\}, \varepsilon \in \mathbb{R}_+$$

Data fitting with square Loss

- Square loss

$$\ell(\hat{y}, y) = (\hat{y} - y)^2$$

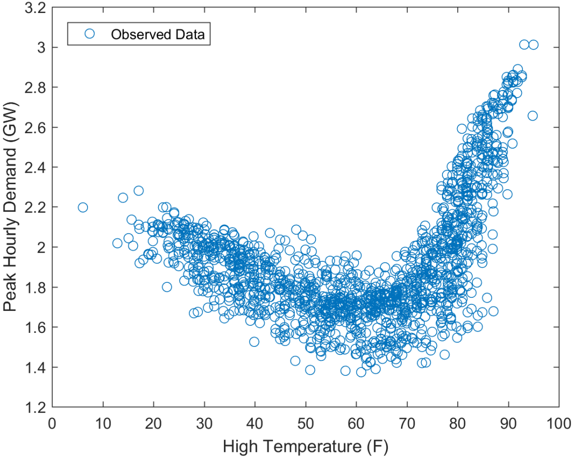
- Called least-squares objective function

$$\underset{\theta}{\text{minimize}}\ (\hat{y} - y)^2 = \underset{\theta}{\text{minimize}}\ \|\hat{y} - y\|_2^2$$

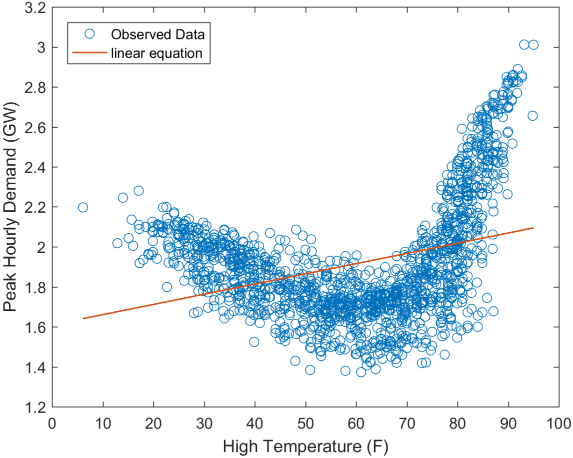
- Has analytical solution

$$\theta^\star = (\Phi^T \Phi)^{-1} \Phi^T y$$

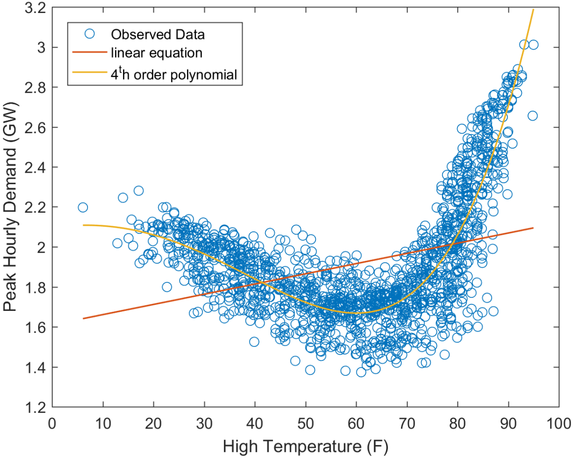
Ex4: Electricity peak demand forecasting



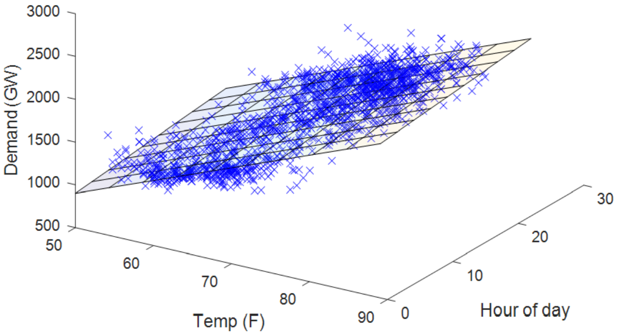
Ex4: Electricity peak demand forecasting



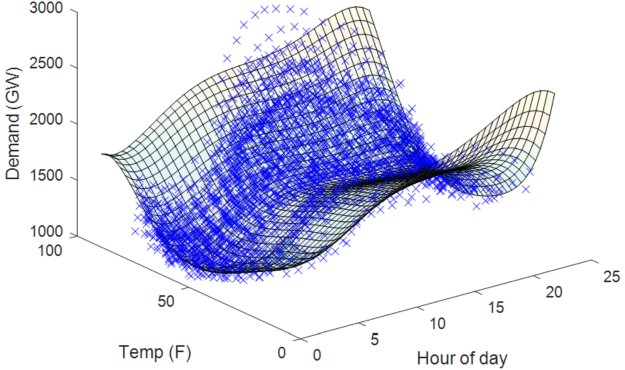
Ex4: Electricity peak demand forecasting



Linear regression 2d vector



"Nonlinear regression" 2d vector



Matlab code

```
X = load('max_temp_b.txt');
y = load('max_demand_b.txt');
[m,n] = size(X);

%% linear
Phi = [X ones(m,1)];
% theta = inv(Phi' * Phi) * Phi' * y;
theta = Phi \ y;

%% 4th order polynomial
Phi = [X.^4 X.^3 X.^2 X ones(m,1)];
theta = Phi \ y;
```

Ex5: Newsvendor problem

- A company produces winter coats.
- The company must commit to specific production quantity x before knowing the exact demand d, 3 months before the winter season.

Ex5: Newsvendor problem

- A company produces winter coats.
- The company must commit to specific production quantity x before knowing the exact demand d, 3 months before the winter season.
- After seeing demand d, the company decides the amount y_r to sell in regular price π_r, and the amount y_s to sell at a salvage/discounted price π_s.
- This is called decision making under *uncertainty*, because decision x is made under uncertain demand d.

Two stage stochastic programming

- Decision variables:
  - Here-and-Now decision: production quantity x
  - Wait-and-See decision: regular price quantity y_r, discounted price quantity y_s
- Objective: minimize production cost and expected future cost

Two stage stochastic programming

- Decision variables:
  - Here-and-Now decision: production quantity x
  - Wait-and-See decision: regular price quantity y_r, discounted price quantity y_s
- Objective: minimize production cost and expected future cost
- Stochastic program

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) + \mathbb{E}_d[Q(x,d)] \\ \text{subject to} \quad & 0 \leq x \leq \hat{x} \end{aligned}$$

Two stage stochastic programming

- Decision variables:
  - Here-and-Now decision: production quantity x
  - Wait-and-See decision: regular price quantity y_r, discounted price quantity y_s
- Objective: minimize production cost and expected future cost
- Stochastic program

$$
\begin{aligned}
\underset{x}{\text{minimize}} \quad & f(x) + \mathbb{E}_d[Q(x, d)] \\
\text{subject to} \quad & 0 \le x \le \hat{x}
\end{aligned}
$$

$$
\begin{aligned}
Q(x, d) = \underset{y_r(d), y_s(d)}{\text{minimize}} \quad & -(\pi_r \; y_r(d) + \pi_s \; y_s(d)) \\
\text{subject to} \quad & y_r(d) \le d, \forall d \in \mathcal{D} \\
& y_r(d) + y_s(d) \le x, \forall d \in \mathcal{D} \\
& y_r(d), y_s(d) \ge 0, \forall d \in \mathcal{D}
\end{aligned}
$$

Reformulation

- Combine the two stages

$$
\begin{array}{ll}
\underset{x,\, y_r(d),\, y_s(d)}{\text{minimize}} & f(x) + \mathbb{E}_d[-(\pi_r\ y_r(d) + \pi_s\ y_s(d))] \\
\text{subject to} & 0 \le x \le \hat{x} \\
& y_r(d) \le d, \forall d \in \mathcal{D} \\
& y_r(d) + y_s(d) \le x, \forall d \in \mathcal{D} \\
& y_r(d), y_s(d) \ge 0, \forall d \in \mathcal{D}
\end{array}
$$

Reformulation

- Combine the two stages

$$
\begin{aligned}
\underset{x,\, y_r(d),\, y_s(d)}{\text{minimize}} \quad & f(x) + \mathbb{E}_d[-(\pi_r\ y_r(d) + \pi_s\ y_s(d))] \\
\text{subject to} \quad & 0 \leq x \leq \hat{x} \\
& y_r(d) \leq d, \forall d \in \mathcal{D} \\
& y_r(d) + y_s(d) \leq x, \forall d \in \mathcal{D} \\
& y_r(d), y_s(d) \geq 0, \forall d \in \mathcal{D}
\end{aligned}
$$

- Suppose demand d is a discrete random variable with S scenarios $(d_1, ..., d_s)$, and each scenario d_i with a probability p_i.
- Correspondingly, the sell quantities have $y_{r,i}$ and $y_{s,i}$ for each scenario d_i.

Reformulation

$$
\begin{array}{ll}
\underset{x,\, y_r(d),\, y_s(d)}{\text{minimize}} & f(x) + \mathbb{E}_d[-(\pi_r \; y_r(d) + \pi_s \; y_s(d))] \\
\text{subject to} & 0 \le x \le \hat{x} \\
 & y_r(d) \le d, \forall d \in \mathcal{D} \\
 & y_r(d) + y_s(d) \le x, \forall d \in \mathcal{D} \\
 & y_r(d), y_s(d) \ge 0, \forall d \in \mathcal{D}
\end{array}
$$

Reformulation

$$\begin{aligned}
\underset{x,\, y_r(d),\, y_s(d)}{\text{minimize}} \quad & f(x) + \mathbb{E}_d[-(\pi_r\ y_r(d) + \pi_s\ y_s(d))] \\
\text{subject to} \quad & 0 \le x \le \hat{x} \\
& y_r(d) \le d, \forall d \in \mathcal{D} \\
& y_r(d) + y_s(d) \le x, \forall d \in \mathcal{D} \\
& y_r(d), y_s(d) \ge 0, \forall d \in \mathcal{D}
\end{aligned}$$

$$\Downarrow$$

$$\begin{aligned}
\underset{x,\, y_{r,i},\, y_{s,i}}{\text{minimize}} \quad & cx - \sum_{i=1}^{S} p_i(\pi_r\ y_{r,i} + \pi_s\ y_{s,i}) \\
\text{subject to} \quad & 0 \le x \le \hat{x} \\
& y_{r,i} \le d_i, \forall i = 1, ..., S \\
& y_{r,i} + y_{s,i} = x, \forall i = 1, ..., S \\
& y_{r,i}, y_{s,i} \ge 0, \forall i = 1, ..., S
\end{aligned}$$

A concrete example

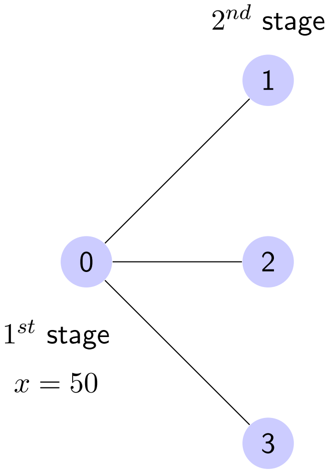
- Suppose there are 3 scenarios, $d_1 = 10$ with probability of $\frac{1}{4}$; $d_2 = 30$ with probability of $\frac{5}{12}$; $d_3 = 50$ with probability of $\frac{1}{3}$.
- Unit cost to produce coats: $c = 5$, regular price: $\pi_r = 10$, discounted price: $\pi_s = 3$.
- Production capacity: $\hat{x} = 70$.

A concrete example

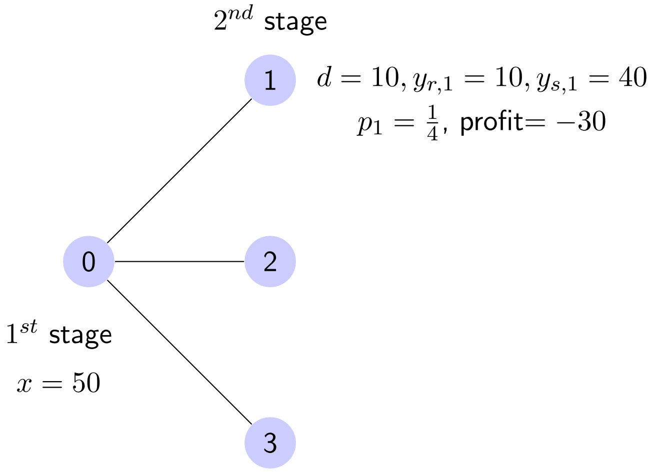
- Suppose there are 3 scenarios, $d_1 = 10$ with probability of $\frac{1}{4}$; $d_2 = 30$ with probability of $\frac{5}{12}$; $d_3 = 50$ with probability of $\frac{1}{3}$.
- Unit cost to produce coats: $c = 5$, regular price: $\pi_r = 10$, discounted price: $\pi_s = 3$.
- Production capacity: $\hat{x} = 70$.

$$\begin{aligned}
\underset{x,\, y_{r,i},\, y_{s,i}}{\text{minimize}} \quad & 5x - [\frac{1}{4}(10y_{r,1} + 3y_{s,1}) + \frac{5}{12}(10y_{r,2} + 3y_{s,2}) \\
& + \frac{1}{3}(10y_{r,3} + 3y_{s,3})] \\
\text{subject to} \quad & 0 \leq x \leq 70 \\
& y_{r,1} \leq 10, y_{r,2} \leq 30, y_{r,3} \leq 50 \\
& y_{r,i} + y_{s,i} \leq x, \forall i = 1, ..., S \\
& y_{r,i}, y_{s,i} \geq 0, \forall i = 1, ..., S
\end{aligned}$$

Results



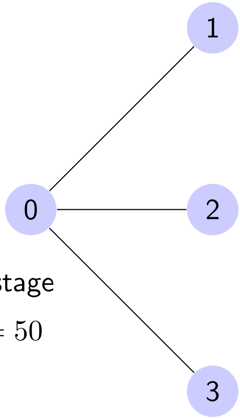
Results



2^{nd} stage
1
$d = 10, y_{r,1} = 10, y_{s,1} = 40$
$p_1 = \frac{1}{4}$, profit$= -30$
0
2
3
1^{st} stage
$x = 50$

Results

2^{nd} stage



$d = 10, y_{r,1} = 10, y_{s,1} = 40$

$p_1 = \frac{1}{4}$, profit$= -30$

$d = 30, y_{r,2} = 30, y_{s,2} = 20$

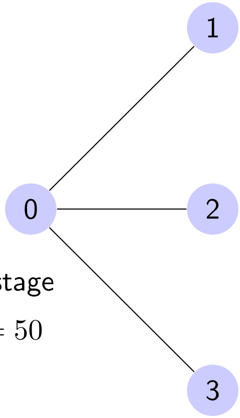
$p_2 = \frac{5}{12}$, profit$=110$

1^{st} stage

$x = 50$

Results

2^{nd} stage



$d = 10, y_{r,1} = 10, y_{s,1} = 40$

$p_1 = \frac{1}{4}$, profit$= -30$

$d = 30, y_{r,2} = 30, y_{s,2} = 20$

$p_2 = \frac{5}{12}$, profit$=110$

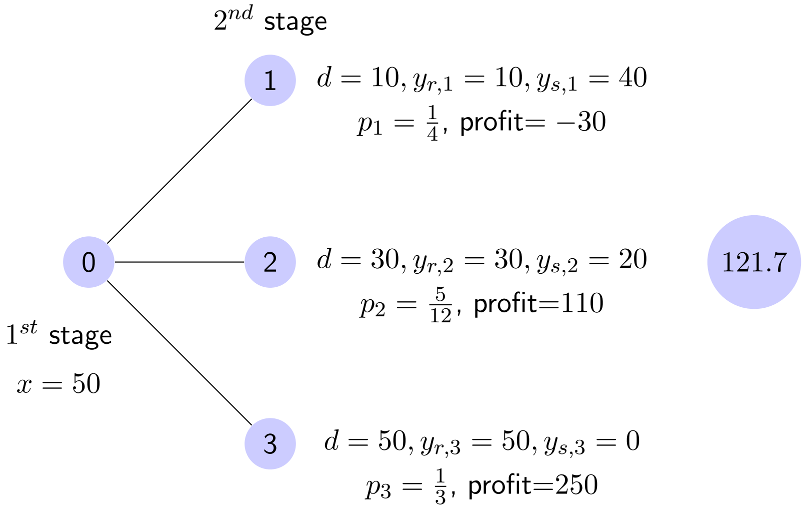
$d = 50, y_{r,3} = 50, y_{s,3} = 0$

$p_3 = \frac{1}{3}$, profit$=250$

1^{st} stage

$x = 50$

Results


2^{nd} stage
1
$d = 10, y_{r,1} = 10, y_{s,1} = 40$
$p_1 = \frac{1}{4}$, profit$= -30$
0
2
$d = 30, y_{r,2} = 30, y_{s,2} = 20$
$p_2 = \frac{5}{12}$, profit$=110$
121.7
1^{st} stage
$x = 50$
3
$d = 50, y_{r,3} = 50, y_{s,3} = 0$
$p_3 = \frac{1}{3}$, profit$=250$

Dealing with the General Case

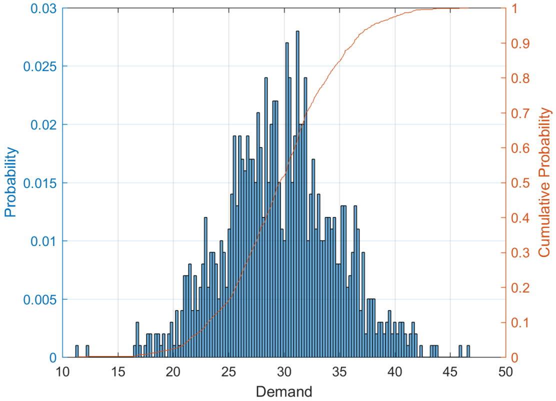
- What if it is hard to define scenarios?

Dealing with the General Case

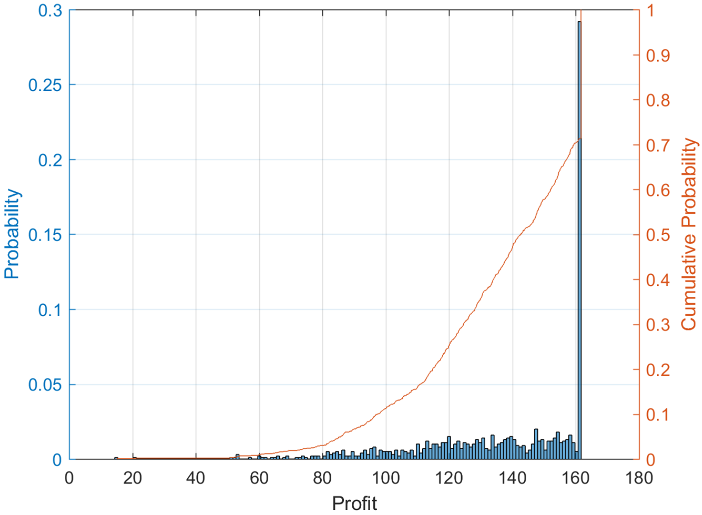
- What if it is hard to define scenarios?
- What if our distribution is not discrete?

Dealing with the General Case

- What if it is hard to define scenarios?
- What if our distribution is not discrete?
- Sampling is enough under most general conditions



Results



0.3
0.25
0.2
0.15
0.1
0.05
0
Probability
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Cumulative Probability
0
20
40
60
80
100
120
140
160
180
Profit

Electricity market-clearing problem

- A two-settlement market clearing problem with day ahead (DA) and real time (RT) stages

$$
\begin{aligned}
\underset{p_g^{DA}, p_{g,\omega}^{RT}, p_{d,\omega}^{shed},}{\text{minimize}} \quad & cost^{DA}(p_g^{DA}) + \mathbb{E}_\omega[cost^{RT}(p_{g,\omega}^{RT}, p_{d,\omega}^{shed})] \\
\text{subject to} \quad & f(p_g^{DA}) \leq 0 \\
& g(p_{g,\omega}^{RT}, p_{d,\omega}^{shed}) \leq 0, \ \forall \omega \\
& g(p_g^{DA}, p_{g,\omega}^{RT}, p_{d,\omega}^{shed}) \leq 0, \ \forall \omega
\end{aligned}
$$

How is the optimization problem actually solved?

- Unconstrained problem. Set the derivative (one dimensional) or gradient (high dimensional)

$$\nabla_x f(x) = 0$$

- Then how to find $\nabla_x f(x) = 0$
  - Direct solution: Analytically compute
  - Gradient descent

$$\text{Repeat: } x \leftarrow x - \alpha \nabla_x f(x)$$

  - Newton's method

$$\text{Repeat: } x \leftarrow x - \left(\nabla_x^2 f(x)\right)^{-1} \nabla_x f(x)$$

Constrained optimization

- Barrier method: Approximate problem via unconstrained optimization

$$\underset{x}{\text{minimize}}\ f(x) - t\sum_{i=1}^{m} log(-g_i(x))$$

as $t \to 0$, this approaches original problem

- Maximize Lagrangian dual problem

$$\underset{\lambda}{\text{maximize}}\ \{\underset{x}{\text{minimize}}\ f(x) + \sum_{i=1}^{m} \lambda_i g_i(x)\}$$

Practically solving optimization problems

- The good news, for many classes of optimization problems, people have already done all the "hard work" of developing numerical algorithms
- A wide range of tools that can take optimization problems in "natural" forms and compute a solution
- Some well-known libraries: CVX (MATLAB), CVXPY (Python), YALMIP(MATLAB), AMPL (custom language), GAMS (custom language), Gurobi (custom language)

Brief history of convex optimization

- Theory (convex analysis): ca1900–1970
- Algorithms
  - 1947: simplex algorithm for linear programming (Dantzig)
  - 1960s: early interior-point methods (Fiacco & McCormick, Dikin, . . .)
  - 1970s: ellipsoid method and other subgradient methods
  - 1980s: polynomial-time interior-point methods for linear programming (Karmarkar 1984)
  - late 1980s–now: polynomial-time interior-point methods for nonlinear convex optimization (Nesterov & Nemirovski 1994)
- Applications
  - before 1990: mostly in operations research; few in engineering
  - since 1990: many new applications in engineering (control, signal
  - processing, communications, circuit design, . . .); new problem classes (semidefinite and second-order cone programming, robust optimization)

MATLAB history

- Invented by Prof. Cleve Moler, University of New Mexico in late 1970s
- The MathWorks, Inc. was formed in 1984 by Moler and Jack little. One product: MATLAB
- Today: 100 products; over 1 million users worldwide.

Take home messages

- Many problems of real importance can be formulated as an optimization problem.
- Particularly, convex optimization problems that can be solved efficiently and which still find a huge number of applications
- Reducing a seemingly new problem to an instance of a well-known problem allows one to use pre-existing methods for solving them

References

C. Bishop, Pattern Recognition and Machine Learning

S. Boyd and L. Vandenberghe, Convex Optimization

Z. Kolter, Computational Methods for Smart Grid

S. Ahmed and A. Sun, Deterministic Optimization at edx.org

DTU CEE Summer School, https://energy-markets-school.dk

D. Espinoza, Solving Simple Stochastic Optimization Problems with Gurobi

Electric Cars, edx.org

Z. Wang, http://uregina.ca/∼wang233z

Thank you!