Optimization

Lin Qingfeng
February 27, 2021

1 Classification of Optimization Problem

- Optimization problems can be classified based on the type of constraints, nature of design variables, physical structure of the problem, nature of the equations involved, deterministic nature of the variables, permissible value of the design variables, separability of the functions and number of objective functions.

1.1 Classification based on existence of constraints.

- **Constrained optimization problems:** which are subject to one or more constraints.
- **Unconstrained optimization problems:** in which no constraints exist.

1.2 Classification based on the nature of the design variables

- **First Category:** the objective is to find a set of design parameters that make a prescribed function of these parameters minimum or maximum subject to certain constraints.
- **Second Category:** the objective is to find a set of design parameters, which are all continuous functions of some other parameter, that minimizes an objective function subject to a set of constraints.

1.3 Classification based on the physical structure of the problem

- **Optimal control problems:** An optimal control (OC) problem is a mathematical programming problem involving a number of stages, where each stage evolves from the preceding stage in a prescribed manner.
- **Non-optimal control problems**

1.4 Classification based on the nature of the equations involved

- Based on the nature of expressions for the objective function and the constraints, optimization problems can be classified as **linear, nonlinear, geometric and quadratic programming problems.**
- **Linear programming problem:** if the objective function and all the constraints are linear functions of the design variables, the mathematical programming problem is called a linear programming (LP) problem.
- **Nonlinear programming problem:** if any of the functions among the objectives and constraint functions is nonlinear, the problem is called a nonlinear programming (NLP) problem this is the most general form of a programming problem.
- **Geometric programming problem:** A geometric programming (GMP) problem is one in which the objective function and constraints are expressed as polynomials in design variables.
- **Quadratic programming problem:** A quadratic programming problem is the best behaved nonlinear programming problem with a quadratic objective function and linear constraints.

1.5 Classification based on the permissible values of the decision variables

- **Integer programming problem:** If some or all of the design variables of an optimization problem are restricted to take only integer (or discrete) values, the problem is called an integer programming problem.
- **Real-valued programming problem:** A real-valued problem is that in which it is sought to minimize or maximize a real function by systematically choosing the values of real variables from within an allowed set. When the allowed set contains only real values, it is called a real-valued programming problem.

1.6 Classification based on deterministic nature of the variables

- **Deterministic programming problem:** In this type of problems all the design variables are deterministic.
- **Stochastic programming problem:** In this type of an optimization problem some or all the parameters (design variables and/or pre-assigned parameters) are probabilistic (non deterministic or stochastic).

1.7 Classification based on separability of the functions

- Based on the separability of the objective and constraint functions optimization problems can be classified as separable and non-separable programming problems.
- **Separable programming problems:** In this type of a problem the objective function and the constraints are separable. A function is said to be separable if it can be expressed as the sum of n single-variable functions.

1.8 Classification based on the number of objective functions

- **Single-objective programming problem:** in which there is only a single objective.
- **Multi-objective programming problem:** in which there are more than one objective.

2 Algorithm

2.1 Unconstrained Optimization

- **Assume all functions are continuous and differentiable.**
- The problem is,

$$\text{minimize} \quad f(x) \tag{1}$$

- Algorithm: **Steepest Descent Method, Newton Method, Quasi-Newton Method**

2.2 Equality Constrained Optimization

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b, \end{array} \tag{2}$$

- $f : \mathbf{R}^n \to \mathbf{R}$ is convex and twice continuously differentiable
- Algorithm: **Eliminating equality constraints, Newton Method**

2.3 Inequality Constrained Optimization

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & Ax = b \end{array} \tag{3}$$

- $f_0, \ldots, f_m : \mathbf{R}^n \to \mathbf{R}$ are convex and twice continuously differentiable
- Algorithm: **Interior-point Methods**

2.4 Difference of Convex Programming

- Difference of convex (DC) programming problems have the form

$$\begin{array}{ll} \text{minimize} & f_0(x) - g_0(x) \\ \text{subject to} & f_i(x) - g_i(x) \leq 0, \quad i = 1, \ldots, m \end{array} \tag{4}$$

- $f_i : \mathbf{R}^n \to \mathbf{R}$ and $g_i : \mathbf{R}^n \to \mathbf{R}$ for $i = 0, \ldots, m$ are convex.
- Algorithm: **Convex-concave programming**

2.5 Integer Constrained Optimization

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & Ax = b \\ & x \in \mathbb{Z} \end{array} \tag{5}$$

- $f_0, \ldots, f_m : \mathbf{R}^n \to \mathbf{R}$ are convex and twice continuously differentiable
- Algorithm: **Branch and Bound Methods**

2.6 ADMM problem

$$\begin{array}{ll} \text{minimize} & f(x) + g(z) \\ \text{subject to} & Ax + Bz = c \end{array} \tag{6}$$

- $f_i : \mathbf{R}^n \to \mathbf{R}$ and $g_i : \mathbf{R}^n \to \mathbf{R}$ for $i = 0, \ldots, m$ are convex.
- Algorithm: **ADMM**

2.7 SCA problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad j = 1, \ldots, p \end{array} \tag{7}$$

- f_0 and f_i (possibly) nonconvex, h_i (possibly) non-affine.
- Algorithm: **Successive Convex Approximation**

2.8 Geometric Programming problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 1, \quad i = 1, \ldots, m \\ & g_i(x) = 1, \quad i = 1, \ldots, p \end{array} \tag{8}$$

- where f_i are posynomial functions, g_i are monomials
- Algorithm: **Successive GP Approximation**

2.9 Combinatorial problem

- Combinatorial optimization is the process of searching for maxima (or minima) of an objective function F whose domain is a discrete but large configuration space (as opposed to an N -dimensional continuous space).

  Simple Examples:

- The Traveling Salesman Problem: given the (x, y) positions of N different cities, find the shortest possible path that visits each city exactly once.
- Job-shop Scheduling: given a set of jobs that must be performed, and a limited set of tools with which these jobs can be performed, find a schedule for what jobs should be done when and with what tools that minimizes the total amount of time until all jobs have been completed.
- Algorithm: **Hungarian Algorithm, Deep Reinforcement Learning**

3 Application

3.1 Estimation Theory

3.1.1 Maximum Likelihood Estimation

- Given the data model,

$$x[n] = r^n + w[n] \quad n = 0, 1, \ldots, N-1 \tag{9}$$

where $w[n]$ is WGN with variance σ^2. The parameter r, the exponential factor, is to be estimated. Allowable values are $r > 0$. The MLE of r is the value that maximizes the likelihood function,

$$p(\mathbf{x}; r) = \frac{1}{(2\pi\sigma^2)^{\frac{N}{2}}} exp\left[-\frac{1}{2\sigma^2}\sum_{n=0}^{N-1}\left(x[n] - r^n\right)^2\right] \tag{10}$$

equivalently, the value that minimizes,

$$J(r) = \sum_{n=0}^{N-1}\left(x[n] - r^n\right)^2 \tag{11}$$

Differentiating $J(r)$ and setting it equal to zero produces,

$$\sum_{n=0}^{N-1}\left(x[n] - r^n\right) nr^{n-1} = 0 \tag{12}$$

we need use iterative algorithm to solve this equation.

3.1.2 The Maximum Entropy

- The maximum entropy (ME) principle states that if we don't know the pdf $f_X(x)$ of X but would like to estimate it with a function, say $p(x)$, a good choice is the function $p(x)$ which maximizes the entropy,

$$H[X] \triangleq -\int_{-\infty}^{\infty} p(x)\ln p(x)dx \tag{13}$$

and which satisfies the constraints

$$\begin{aligned} p(x) &\geq 0 \\ \int_{-\infty}^{\infty} p(x)dx &= 1 \\ \int_{-\infty}^{\infty} xp(x)dx &= \mu \\ \int_{-\infty}^{\infty} x^2 p(x)dx &= m_2, \text{ and so forth.} \end{aligned} \tag{14}$$

Suppose we know from measurements or otherwise only μ. Using the method of Lagrange multipliers, the solution is obtained by maximizing the expression,

$$-\int_0^{\infty} p(x)\ln p(x)dx - \lambda_1\int_0^{\infty} p(x)dx - \lambda_2\int_0^{\infty} xp(x)dx \tag{15}$$

After differentiating we obtain,

$$p(x) = \begin{cases} \frac{1}{\mu} e^{-x/\mu}, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{16}$$

The problem of obtaining the ME estimate of $f_X(x)$ when both μ and σ^2 are known is left as an exercise. In this case $p(x)$ is the Normal distribution with mean μ and variance σ^2.

3.2 Machine Learning Problem

3.2.1 Support Vector Machine(SVM)

- **Algorithm: Lagrange Multiplier Method, Sequential Minimal Optimization**
- **Problem Formulation:**

$$\begin{gathered} \min_{w,b} \frac{1}{2} w^T w \\ \text{s.t. } y_i \left(w^T x_i + b \right) \geq 1, i = 1, 2, \ldots, N \end{gathered} \tag{17}$$

Lagrange Function:

$$L(w, b, \alpha) = \frac{1}{2} w^T w + \sum^{N} \alpha_i \left(1 - y_i \left(w^T x_i + b \right) \right) \tag{18}$$

Dual Problem:

$$\begin{gathered} \min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i=1}^{N} \alpha_i \\ \text{s.t. } \sum_{i=1}^{N} \alpha_i y_i = 0 \\ \alpha_i \geq 0, i = 1, 2, \ldots, N \end{gathered} \tag{19}$$

3.2.2 Gaussian Mixture Model

- **Algorithm: EM Algorithm**
- **Problem Formulation:**

$$\begin{aligned} \Theta_{MLE} &= \arg\max_{\Theta} \mathcal{L}(\Theta \mid X) \\ &= \arg\max_{\Theta} \left(\sum_{i=1}^{n} \log \sum_{l=1}^{k} \alpha_l \mathcal{N}\left(X | \mu_l, \Sigma_l\right) \right) \end{aligned} \tag{20}$$

- For each iteration if the E-M algorithm, we perform:

$$\Theta^{(g+1)} = \arg\max_{\theta} \left(\int_z \log(p(X, Z|\theta)) p\left(Z | X, \Theta^{(g)} \right) \right) dz \tag{21}$$

3.2.3 The discrete-time Kalman Filter

- **Algorithm: Unconstrained Optimization Algorithm**
- **Model:** The dynamic system is given by the following equations:

$$\begin{aligned} x_k &= F_{k-1} x_{k-1} + G_{k-1} u_{k-1} + w_{k-1} \\ y_k &= H_k x_k + v_k \\ E\left(w_k w_j^T \right) &= Q_k \delta_{k-\jmath} \\ E\left(v_k v_j^T \right) &= R_k \delta_{k-\jmath} \\ E\left(w_k v_j^T \right) &= 0 \end{aligned} \tag{22}$$

3.3 Communication Problem

3.3.1 Algorithm: AM Framework + First-Order Algorithm

- **Problem: Content-Sparse Multicast Beamforming in Large-Scale C-RAN**
  Problem Formulation:

$$\begin{aligned}\min_{\mathcal{W}} \; & \underbrace{\sum_{m=1}^{M}\sum_{n=1}^{N}\left\|\|\mathbf{w}_{m,n}\|_2^2\right\|_0 (1-c_{f_m,n})\,R_m}_{\text{backhaul cost}} \\ & + \underbrace{\eta\sum_{m=1}^{M}\sum_{n=1}^{N}\|\mathbf{w}_{m,n}\|_2^2}_{\text{transmit power cost}} \\ \text{s.t. } & \frac{|\mathbf{h}_k^H\mathbf{w}_{m_k}|^2}{\sum_{m\neq m_k}|\mathbf{h}_k^H\mathbf{w}_m|^2+\sigma_k^2} \geq \gamma_{m_k}, \quad \forall k=1,2,\ldots,K, \end{aligned} \tag{23}$$

  The challenge in solving this problem,

- the discontinuity/non-convexity of the cost function and the non- convexity of the constraints
- the challenges in the cost function and the K constraints are coupled as they all depend on the same variable set $\mathcal{W}$.

  Method 1:
  One solution is to approximate the discontinuous cost function by a smooth function and then apply convex-concave procedure (CCP) to solve a sequence of non-trivial SOCP subproblems with the interior-point method.

  Limitation:
  Such an approach does not decouple the cost function and the constraints, making its complexity order cubic with respect to the problem size. When the scale is large, this method is invalid.

  Method 2:
  The First step: Introduce auxiliary variables, The problem becomes,

$$\begin{aligned}\min_{\mathcal{W},\mathcal{V}} \; & \sum_{m=1}^{M}\sum_{n=1}^{N}\left\|\|\mathbf{w}_{m,n}\|_2^2\right\|_0 \alpha_{m,n} + \eta\sum_{m=1}^{M}\sum_{n=1}^{N}\|\mathbf{w}_{m,n}\|_2^2 \\ & + \rho\sum_{m=1}^{M}\sum_{k=1}^{K}\left|v_{k,m}-\mathbf{h}_k^H\mathbf{w}_m\right|^2 \\ \text{s.t. } & \gamma_{m_k}\left(\sum_{m\neq m_k}|v_{k,m}|^2+\sigma_k^2\right)-|v_{k,m_k}|^2 \leq 0, \\ & \qquad\qquad \forall k=1,2,\ldots,K, \end{aligned} \tag{24}$$

  The Second Step: Fixing $\mathcal{W}$, tackle the QCQP problem by using KKT condition.

  The Third Step: Fixing $\mathcal{V}$, tackle the discontinuous nonconvex problem by using SCA and subgradient algorithm.

3.3.2 Algorithm: Branch and Bound Framework + Accelerate Algorithm

- **Problem: Channel Selection and Power Allocation Problem in Communication**

  Problem Formulation:

$$
\begin{aligned}
J_1: \ \{P_m^c, P_k^v, \rho_{m,k}\} = & \\
& \max_{\{P_m^c, P_k^v, \rho_{m,k}\}} \min_{\{k \in \mathcal{K}\}} C_k^v \\
& s.t. \ 0 \le P_m^c \le P_{max}^c, \ \forall m \in \mathcal{M} \\
& \quad 0 \le \sum_{m \in \mathcal{M}} \rho_{m,k} P_{m,k}^v \le P_{max}^v, \ \forall k \in \mathcal{K} \\
& \quad \sum_{k \in \mathcal{K}} \rho_{m,k} \le 1, \ \forall m \in \mathcal{M} \\
& \quad \rho_{m,k} \in \{0,1\}, \ \forall m \in \mathcal{M}, k \in \mathcal{K} \\
& \quad \log_2(1+\gamma_m^c) \ge \gamma_0^c, \ \forall m \in \mathcal{M}
\end{aligned} \tag{25}
$$

  The challenge in solving this problem

- Resource allocation problem in communication s is usually formulated as mixed integer nonlinear programming (MINLP) problems, which are in general NP-hard and no efficient global optimal algorithm is available yet.
- In general, the optimal solutions to the MINLP problems can be only achieved by the branch-and-bound algorithm. However, the worst-case computational complexity of the B&B algorithm is exponential causing it impractical for real-time implementation.

  Method:

  The First step: Using branch and bound method to solve the original problem,

$$
\begin{aligned}
J_4: \ \{t, s_{m,k}, \rho_{m,k}\} = & \max_{\{t, s_{m,k}, \rho_{m,k}\}} t \\
& 0 \le \rho_{m,k} \le 1, \forall k \in \mathcal{K} \ \ \forall m \in \mathcal{M} \\
& \sum_{k \in \mathcal{K}} \rho_{m,k} \le 1, \ \forall m \in \mathcal{M}, \\
& 0 \le \sum_{m \in \mathcal{M}} s_{m,k} \le P_{max}^v, \ \forall k \in \mathcal{K} \\
& \sum_{m \in \mathcal{M}} \rho_{m,k} \log_2(1 + \frac{s_{m,k}}{a_{m,k}\rho_{m,k} + b_{m,k} s_{m,k}}) \ge t \ \forall k \in \mathcal{K}
\end{aligned} \tag{26}
$$

  The second step: Using **Machine Learning Method** to accelerate this BB algorithm, for example, reinforcement learning, imitation learning and so on.

  There are also some sub-optimal method,

- **Game theory**
- **Graph theory**
- **Heuristic algorithms**

  Limitation:

- the performance gaps between the sub-optimal solutions and the optimal ones are hard to control.
- many iterative based heuristic algorithms still have high computational complexity for real-time implementation.

3.3.3 Algorithm: Deep Reinforcement Learning Framework

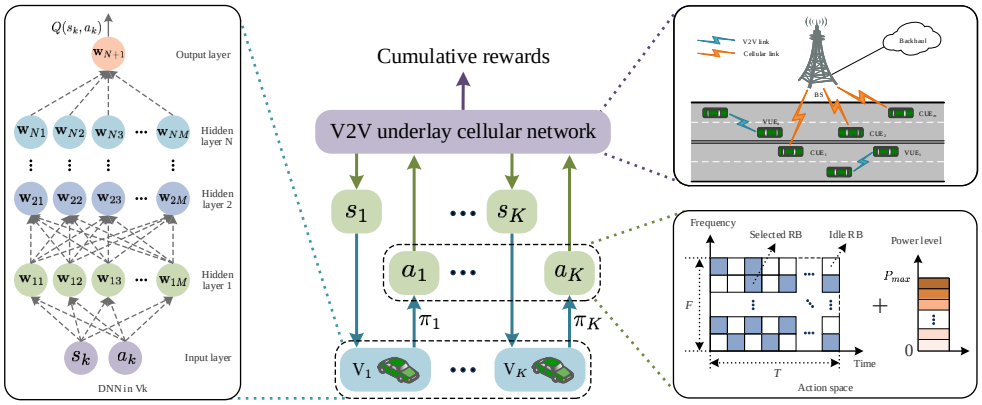

Figure 1: resource allocation in V2V networks

- **Problem: Resource Allocation Problem in V2V networks**
  Problem Formulation:

$$
\begin{aligned}
&\{P_m^c, P_k^v, \rho_{m,k}\} = \\
&\max_{\{P_m^c, P_k^v, \rho_{m,k}\}} \min_{\{k \in \mathcal{K}\}} C_k^v \\
&s.t.\ 0 \le P_m^c \le P_{max}^c,\ \forall m \in \mathcal{M} \\
&\quad 0 \le \sum_{m \in \mathcal{M}} \rho_{m,k} P_{m,k}^v \le P_{max}^v,\ \forall k \in \mathcal{K} \\
&\quad \sum_{k \in \mathcal{K}} \rho_{m,k} \le 1,\ \forall m \in \mathcal{M} \\
&\quad \rho_{m,k} \in \{0,1\},\ \forall m \in \mathcal{M}, k \in \mathcal{K} \\
&\quad \log_2(1+\gamma_m^c) \ge \gamma_0^c,\ \forall m \in \mathcal{M}
\end{aligned} \tag{26}
$$

Traditional Method:

We can use branch and bound method to solve this mixed integer non linear programming problem. However, this method will be invalid when the scale of the problem gets large.

Deep Reinforcement Method:
The key of deep reinforcement learning(DRL) is to determine state, action and reward of agents.

- state: constant in communication problem
- action: design variables
- reward: the value of objective function

According to the nature of design variables, we can selection the algorithm. If the design variable is discrete, we usually choose deep Q networks. And if the variable is continuous, we choose deep deterministic policy gradient.

3.3.4 Algorithm: Majorization-Minimization Framework

- **The MM procedure consists of two steps.**
- In the first majorization step, we find a surrogate function that locally approximates the objective function with their difference minimized at the current point. In other words, the surrogate upperbounds the objective function up to a constant.
- Then in the minimization step, we minimize the surrogate function. The procedure is shown pictorially in Figure. 2.
- **The key to the success of MM lies in constructing a surrogate function. Generally speaking, surrogate functions with the following features are desired:**
- Separability in variables (parallel computing);
- Convexity and smoothness;
- The existence of a closed-form minimizer.
- **The MM Algorithm**
  The Problem Formulation:

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & \mathbf{x} \in \mathcal{X} \end{aligned} \tag{27}$$

Initialized as $\mathbf{x}_0 \in \mathcal{X}$, MM generates a sequence of feasible points $(\mathbf{x}_t)_{t\in\mathbb{N}}$ by the following induction. At point $\mathbf{x}_t$, in the majorization step we construct a continuous surrogate function $g(\cdot \mid \mathbf{x}_t) : \mathcal{X} \to \mathbb{R}$ satisfying the upperbound property that

$$g(\mathbf{x} \mid \mathbf{x}_t) \geq f(\mathbf{x}) + c_t, \forall \mathbf{x} \in \mathcal{X} \tag{28}$$

Then in the minimization step, we update $\mathbf{x}$ as

$$\mathbf{x}_{t+1} \in \arg\min_{\mathbf{x}\in\mathcal{X}} g(\mathbf{x} \mid \mathbf{x}_t) \tag{29}$$

4 Important Operation in Optimization

4.1 Introduce Slack Variable

4.2 SCA: construct surrogate function

- **First Order Taylor Expansion**
  Suppose f can be decomposed as

$$f(\mathbf{x}) = f_0(\mathbf{x}) + f_{ccv}(\mathbf{x}) \tag{30}$$

where f_{ccv} is a differentiable concave function. Linearizing f_{ccv} at $\mathbf{x} = \mathbf{x}_t$ yields the following inequality:

$$f_{ccv}(\mathbf{x}) \leq f_{ccv}(\mathbf{x}_t) + \nabla f_{ccv}(\mathbf{x}_t)^T (\mathbf{x} - \mathbf{x}_t) \tag{31}$$

thus f can be upper bounded as

$$f(\mathbf{x}) \leq f_0(\mathbf{x}) + \nabla f_{\text{ccv}}(\mathbf{x}_t)^T \mathbf{x} + \text{ const.} \tag{32}$$

- **Convexity Inequality** For a convex function f_{cvx}, we have the following inequality:

$$f_{cvx}\left(\sum_{i=1}^{n} w_i \mathbf{x}_i\right) \leq \sum_{i=1}^{n} w_i f_{cvx}\left(\mathbf{x}_i\right) \tag{33}$$

- **Construction by Second Order Taylor Expansion**

  If f has bounded curvature, i.e., there exists a matrix $\mathbf{M}$ such that $\mathbf{M} \succeq \nabla^2 f(\mathbf{x}), \forall \mathbf{x} \in \mathcal{X}$, then the following inequality implied by Taylor's theorem holds:

$$f(\mathbf{x}) \leq f(\mathbf{y}) + \nabla f(\mathbf{y})^T(\mathbf{x}-\mathbf{y}) + \frac{1}{2}(\mathbf{x}-\mathbf{y})^T\mathbf{M}(\mathbf{x}-\mathbf{y}) \tag{34}$$

- **Schur Complement**

  The Schur complement condition for $\mathbf{C} \succ \mathbf{0}$ states that

$$\mathbf{X} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \succeq \mathbf{0} \tag{35}$$

  if and only if the Schur complement of $\mathbf{C}$ is in $\mathbb{S}_+$. That is,

$$\mathbf{S} := \mathbf{A} - \mathbf{B}\mathbf{C}^{-1}\mathbf{B}^T \succeq \mathbf{0}. \tag{36}$$