Probabilistic Algorithms

Paul Cotofrei

information management institute

PA 2016

Outline

Motivation and supporting results

- Multiple Objective Functions
- No Free Lunch theorem

Some problems asking optimization

- Find the best red-yellow-green signal timings in an urban traffic network
- Determine the optimal schedule for use of laboratory facilities to serve an organization's overall interests
- Minimize the costs of shipping from production facilities to warehouses
- Maximize the probability of detecting an incoming warhead (vs. decoy) in a missile defense system
- Place sensors in manner to maximize useful information
- Determine the times to administer a sequence of drugs for maximum therapeutic effect

Model the problems as a mathematical model depending on:

1. a set of adjustable parameters/variables
2. an (objective) function defined on the set of parameters
3. a goal: minimize/maximize the function

Two Fundamental Problems of Interest

- Let **D** be the domain of allowable values for a vector **d**
- **d** represents a vector of "adjustables" and may be continuous or discrete (or both)

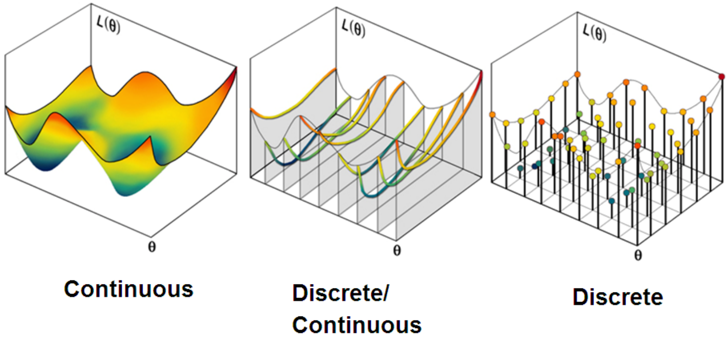
Two fundamental problems of interest

- **Problem 1.** Find the value(s) of a vector $\mathbf{d} \in \mathbf{D}$ that minimize a scalar-valued *loss function* $L(\mathbf{d})$
- **Problem 2.** Find the value(s) of $\mathbf{d} \in \mathbf{D}$ that solve the equation $g(\mathbf{d}) = 0$ for some vector-valued function $g(\mathbf{d})$

Frequently (but not necessarily) $g(\mathbf{d}) = \partial L(\mathbf{d})/\partial \mathbf{d}$

Convert P1 => P2 ? Convert P2 => P1?

Three Common Types of Loss Functions



Forms of loss function

- Mathematical expression
- Algorithms (simulations)
- Physical experiments

Classical Calculus-Based Optimization

- Classical optimization setting of interest

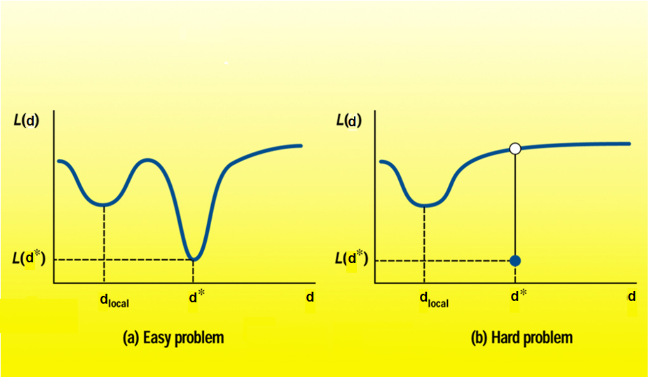
$$\mathbf{D}^* \equiv \min_{\mathbf{d} \in \mathbf{D}} L(\mathbf{d}) = \{\mathbf{d}^* \in \mathbf{D} : L(\mathbf{d}^*) \leq L(\mathbf{d}) \text{ for all } \mathbf{d} \in \mathbf{D}\}$$

  where $\mathbf{d}$ is a n-dimensional vector of parameters and $\mathbf{D} \in \mathcal{R}^n$ is the domain representing the constraints on allowable values for $\mathbf{d}$.

- $\mathbf{D}^*$: a single point, a countable collection of points or an uncountable number
  1. $L(\mathbf{d}) = \mathbf{d}^T\mathbf{d}$, $\mathbf{D} = R^n$; $\mathbf{D}^* =?$
  2. $L(\mathbf{d}) = sin(\mathbf{d})$, $\mathbf{D} = [0, 4\pi]$; $\mathbf{D}^* =?$
  3. $L(\mathbf{d}) = cos(\mathbf{d})$, $\mathbf{D} = R$; $\mathbf{D}^* =?$
  4. $L(\mathbf{d}) = (\mathbf{d}^T\mathbf{d} - 1)^2$, $\mathbf{D} = R^n$; $\mathbf{D}^* =?$

Global vs. Local Solutions

- Any $\mathbf{d}^* \in \mathbf{D}^*$ is a *global* solution
- A *local* solution $\mathbf{d}_{local}$ satisfies $L(\mathbf{d}_{local}) \leq L(\mathbf{d})$ for any $\mathbf{d}$ in a vicinity of $\mathbf{d}_{local}$



Global vs. Local Methods

- General global optimization problem is very difficult
- Sometimes local optimization is "good enough" given limited resources available
- Global methods include: genetic algorithms, evolutionary strategies, simulated annealing, etc.
- Global methods tend to have following characteristics:
  - Inefficient, especially for high-dimensional **d**
  - Relatively difficult to use (e.g., require very careful selection of algorithm coefficients)
  - Sometimes questionable theoretical foundation for global convergence
  - Multiple runs usually required to have confidence in reaching global optimum

Stochastic Optimization

A. Random noise in input information (e.g., measurements with noise for $L(\mathbf{d})$ or $g(\mathbf{d})$):

$$y(\mathbf{d}) \equiv L(\mathbf{d}) + \varepsilon(\mathbf{d})$$

$$Y(\mathbf{d}) \equiv g(\mathbf{d}) + e(\mathbf{d}),$$

where ε and e represent the noise terms.
B. Injected randomness (Monte Carlo) in choice of algorithm iteration magnitude/direction

- Contrasts with deterministic methods (e.g., steepest descent, Newton-Raphson, etc.)
  - Assume perfect information about $L(\mathbf{d})$ (and its gradients)
  - Search magnitude/direction deterministic at each iteration
- Injected randomness (B) in search magnitude/direction can offer benefits in efficiency and robustness
  - E.g., Capabilities for global (vs. local) optimization

General Structure of Optimization Process

Concepts and Definitions

- **Problem Space**: the set $\mathbf{D}$ containing all elements $\mathbf{d}$ which could be the solution of an optimization problem related to a *loss function L*.
  - For the same optimization problem, different problem spaces can be defined
  - The problem space can be restricted by *logical constraints* or *practical constraints*
- **Solution candidate**: an element $\hat{\mathbf{d}}$ of the problem space $\mathbf{D}$ for a certain optimization problem.
- **Solution space**: the set $\mathbf{D}^*$ of all solutions of an optimization problem.
- **Search space**: the set $\mathcal{G}$ of all elements $\hat{\mathbf{d}} \in \mathbf{D}$ which can be processed by an optimization algorithm in order to solve a given problem.
- **Search operations**: the operations used by optimization algorithms in order to explore the search space $\mathcal{G}$.

Outline

Motivation and supporting results

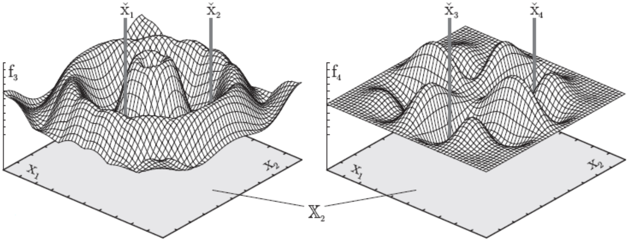
- Multiple Objective Functions
- No Free Lunch theorem

Multiple Objective Functions

- For many real-world design or decision making problems: optimize the set F of m criterions:

$$F = \{L_i : \mathbf{D}_i \mapsto Y_i : 0 < i \leq m, Y_i \subseteq \mathbb{R}\}$$

- Example:
  1. *Maximize* profit and *Minimize* costs for advertising, personal, raw materials etc..
  2. *Maximize* product quality and *Minimize* negative impact on environment.
- Minimize $L_1 : \mathbb{R}^2 \mapsto \mathbb{R}$ and $L_2 : \mathbb{R}^2 \mapsto \mathbb{R}$



Two functions $\mathbf{L_1}$ and $\mathbf{L_2}$ with different minima $\mathbf{x_1}$, $\mathbf{x_2}$, $\mathbf{x_3}$, and $\mathbf{x_4}$.

Approaches for optimum definition

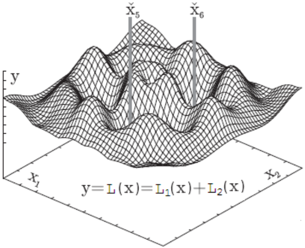
- **Weighted Sums** (or Linear Aggregation)

$$\text{Minimize } L(\mathbf{d}) = \sum_{i=1}^{n} \omega_i w_i L_i(\mathbf{d})$$

where $\omega_i = \begin{cases} 1 & \text{if } L_i \text{ should be minimized} \\ -1 & \text{if } L_i \text{ should be maximized} \end{cases}$

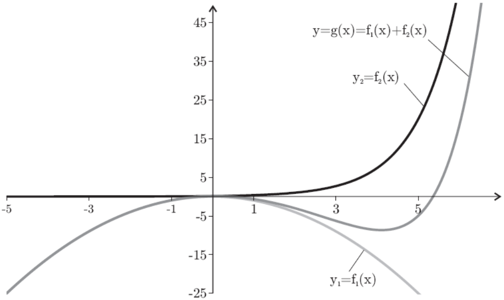
and w_i is the relative importance of L_i

- Exemple: Consider equal relative importance w_1 and w_2 for L_1 and L_2



Drawbacks of linear aggregation

- How to determine the weights w_i ?
- Not suitable for functions from different O classes.
  - Exemple: $f_1(x) = x^2$ and $f_2(x) = e^{x-2}$, so $f_1(x) = O(x^2) \neq O(e^x) = f_2(x)$
  - For x around 0, f_2 is negligible compared to f_1; for $x > 5$, f_1 is negligible compared to f_2.



Pareto Optimization

- It is based on the concept of **Pareto domination**: an element $\mathbf{d}_1$ dominates (is preferred to) an element $\mathbf{d}_2$ (i.e. $\mathbf{d}_1 \vdash \mathbf{d}_2$) if $\mathbf{d}_1$ is better than $\mathbf{d}_2$ in at least one objective function and not worse with respect to all other objectives.

$$\mathbf{d}_1 \vdash \mathbf{d}_2 \Leftrightarrow \begin{cases} \forall i \in 1..n,\ \omega_i L_i(\mathbf{d}_1) \leq \omega_i L_i(\mathbf{d}_2), \textit{and} \\ \exists j \in 1..n : \omega_j L_j(\mathbf{d}_1) < \omega_j L_j(\mathbf{d}_2) \end{cases}$$

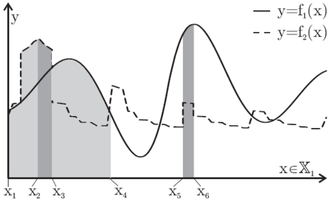
  where

$$\omega_i = \begin{cases} 1 & \text{if } L_i \text{ should be minimized} \\ -1 & \text{if } L_i \text{ should be maximized} \end{cases}$$

- **Pareto optimal**: An element $\mathbf{d} \in \mathbf{D}$ is Pareto optimal if it is not dominated by any other element in the problem space $\mathbf{D}$.
- The optimal set $\mathbf{D}^* = \{\mathbf{d}^* \in \mathbf{D} |\ \nexists \mathbf{d} \in \mathbf{D} : \mathbf{d} \vdash \mathbf{d}^*\}$

Pareto Optimal

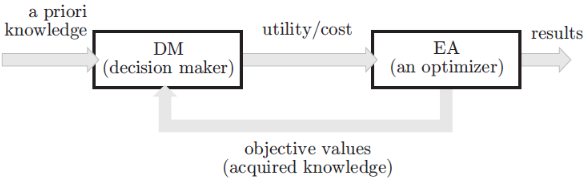
Example: maximize $f_1(x)$ and $f_2(x)$ on $\mathbf{D} = [0, \infty)$



- $f_1(x_2) > f_1(x)$ and $f_2(x_2) > f_2(x)$ for all $x \in [x_1, x_2)$, so $x_2 \vdash x$ $\forall x \in [x_1, x_2)$.
- The points $x \in [x_2, x_3]$ are not dominated by any other points, so are Pareto optimal.
- $f_1(x_5) > f_1(x)$ and $f_2(x_5) > f_2(x)$ for all $x \in [x_3, x_4)$, so $x_5 \vdash x$ $\forall x \in [x_3, x_4)$.
- The points $x \in [x_5, x_6]$ are not dominated by any other points, so are Pareto optimal.
- The set $\mathbf{D}^*$ of Pareto optimal points is $[x_2, x_3] \cup [x_5, x_6]$

External Decision Maker

- A weakness of Pareto optimization: one may have two elements, $\mathbf{d}_1$ and $\mathbf{d}_2$ such that neither $\mathbf{d}_1 \vdash \mathbf{d}_2$ nor $\mathbf{d}_2 \vdash \mathbf{d}_1$.
- For many optimization problems we need a total order: the solution $\mathbf{d}_1$ is better, equal or worse than solution $\mathbf{d}_2$
- A total order using Pareto optimization : *Pareto ranking*
  - In the first step, the elements not dominated receive rank 0
  - In the following steps, one removes the elements with rank i from $\mathbf{D}$, and the new non-dominated elements receive rank $i+1$
- The **External Decision Maker**: it uses a-priory knowledge and user preference to provide a cost function $u : \mathbf{Y} \mapsto R$ which maps the space of objective values to the space of real numbers (total order).



Outline

Motivation and supporting results

- Multiple Objective Functions
- No Free Lunch theorem

No Free Lunch Theorems

- Wolpert and Macready (1997) establish several "No Free Lunch" (NFL) Theorems for optimization
- NFL Theorems apply to settings where parameter set **D** and set of loss function values are finite, discrete sets
  - Relevant for continuous **d** problem when considering digital computer implementation
  - Results are valid for deterministic and stochastic settings
- Number of optimization problems - mappings from **D** to set of loss values - is finite
- NFL Theorems state, in essence, that no one optimisation algorithm is "best" for all problems

No Free Lunch Theorems - Basic Formulation

- Suppose that

$$\begin{aligned} N_{\mathbf{d}} &= \text{ number of values of } \mathbf{d} \\ N_L &= \text{ number of values of loss function} \end{aligned}$$

- Then

$$(N_L)^{N_{\mathbf{d}}} = \text{ number of loss functions}$$

- There is a finite (but possibly huge) number of loss functions
- Performance measure: the lowest value $L(\hat{\theta})$ obtained after k distinct evaluations of loss function L (i.e. $L(\hat{\mathbf{d}}_1), .., L(\hat{\mathbf{d}}_k)$)
- Basic form of NFL considers average performance over all loss functions

Illustration of No Free Lunch Theorems

- Three values of **d**, two outcomes for noise free loss *L*
  - Eight possible mappings, hence eight optimization problems
- Mean loss across all problems is same regardless of **d**; entries 1 or 2 in table below represent two possible *L* outcomes

$\mathbf{d}\backslash$Map	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
$\mathbf{d}_1$	1	1	1	2	2	2	1	2
$\mathbf{d}_2$	1	1	2	1	1	2	2	2
$\mathbf{d}_3$	1	2	2	1	2	1	1	2

Overall Consequences of NFL Theorems

- NFL Theorems state, in essence, that

  Averaging (uniformly) over all possible problems (loss functions L), all algorithms perform equally well

- In particular, if algorithm 1 performs better than algorithm 2 over some set of problems, then algorithm 2 performs better than algorithm 1 on another set of problems

  Overall relative efficiency of two algorithms cannot be inferred from a few sample problems

- NFL theorems say nothing about specific algorithms on specific problems