

Problem Solving

CS6240 Multimedia Analysis

Leow Wee Kheng

Department of Computer Science
School of Computing
National University of Singapore

Problem Solving

First, you **MUST** know **what** is the ideal solution.

It's like climbing a mountain.



- You **MUST** know where's the **destination**.
- Then, you figure out **where** you are and **how** to go there.

Problem definition states

- the **what** part, which includes
 - the initial condition (**where** you are) and
 - the goal condition (**what/where** is the destination),

Algorithm states

- the **how** part, which includes
 - the inputs
 - the outputs
 - the computational steps

A Practical Example: Image Registration

Recall the problem definition:

Given a set S of n points \mathbf{x}_i and a set S' of n points \mathbf{x}'_i , determine the function $f : S \rightarrow S'$ that minimizes the sum-squared error E :

$$E = \sum_{i=1}^n \|\mathbf{A}\mathbf{p}_i - \mathbf{p}'_i\|^2 \quad (1)$$

where $\|\cdot - \cdot\|$ denotes Euclidean difference.

For simplicity, suppose \mathbf{A} is an **affine** transformation matrix.

How to solve this problem?

A straightforward technique is to apply **high-school technique**:
function minimization method.

- Differentiate E with respect to \mathbf{A} .
- Solve for the \mathbf{A} such that $\partial E / \partial \mathbf{A} = 0$.

$$\frac{\partial E}{\partial \mathbf{A}} = -2 \sum_i (\mathbf{p}'_i - \mathbf{A} \mathbf{p}_i) \mathbf{p}_i^T = 0 \quad (2)$$

$$\sum_i \mathbf{A} \mathbf{p}_i \mathbf{p}_i^T = \sum_i \mathbf{p}'_i \mathbf{p}_i^T$$

That is,

$$\sum_i \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} [x_i \ y_i \ 1] = \sum_i \begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} [x_i \ y_i \ 1] \quad (3)$$

Rearranging terms yield

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \mathbf{a} = \mathbf{b} \quad (4)$$

where

$$\mathbf{M} = \begin{bmatrix} \sum_i x_i^2 & \sum_i x_i y_i & \sum_i x_i \\ \sum_i x_i y_i & \sum_i y_i^2 & \sum_i y_i \\ \sum_i x_i & \sum_i y_i & \sum_i 1 \end{bmatrix}$$

$$\mathbf{0} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{a} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{21} & a_{22} & a_{23} \end{bmatrix}^T$$

$$\mathbf{b} = \begin{bmatrix} \sum_i x'_i x_i & \sum_i x'_i y_i & \sum_i x'_i & \sum_i y'_i x_i & \sum_i y'_i y_i & \sum_i y'_i \end{bmatrix}^T.$$

Now, we have a system of linear equation

$$\mathbf{F} \mathbf{a} = \mathbf{b} \tag{5}$$

where

$$\mathbf{F} = \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{bmatrix} \tag{6}$$

which can be solved using standard technique.

For example, pseudo inverse

$$\mathbf{a} = (\mathbf{F}^\top \mathbf{F})^{-1} \mathbf{F}^\top \mathbf{b}. \tag{7}$$

General Problem Solving Approach

- First, define the problem.
That means, you **must** know what is the ideal solution.
- Then, imagine you have infinite space and time and knowledge, determine the simplest algorithm that can produce the ideal solution.
- In real life, you don't have infinite space and time and knowledge. So, determine an efficient algorithm that can give the best approximation to the ideal solution.
- It is often useful to divide a problem into smaller parts that can be solved more easily.

Solving Problems by Optimization

- Many multimedia analysis problems can be formulated as optimization problems.
- However, this does not mean that you can only solve the problem by brute-force algorithms such as exhaustive search or greedy algorithms.

Reasons:

- Real problems have many many parameters to be determined, i.e., high dimensionality problem.
- Real problems have many many locally optimal solutions.
- Study the problem well, and identify useful conditions or constraints.
- Then, apply appropriate algorithms.

Types of Optimization Algorithms

Two broad categories of optimization algorithms:

① Continuous Space Algorithms:

- Optimize in continuous parameter space.
- Examples: least-squares method (Eq. 7), gradient descent, conjugate gradient, quasi-Newton method, Levenberg-Marquadt, Expectation Maximization (EM), linear programming, quadratic programming, simulated annealing, monte-carlo.
- Will illustrate some of them in this course.

② Discrete Space Algorithms:

- Optimize in discrete parameter space.
- Examples: dynamic programming, integer linear programming, graph-theoretic algorithms, heuristic search, simulated annealing, etc.
- For details, refer to CS5206 Foundations in Algorithms, CS5234 Combinatorial and Graph Algorithms, CS6234 Advanced Algorithms.

Another categorization of optimization algorithms:

- ① Unconstrained optimization algorithms:
 - For solving unconstrained optimization problems.
 - Examples: linear least square, gradient descent, conjugate gradient, quasi-Newton method, Levenberg-Marquadt, simulated annealing, monte-carlo.
- ② Constrained optimization algorithms:
 - For solving constrained optimization problems.
 - Examples: Lagrange multiplier, penalty method, equality-constrained method, Expectation Maximization (EM), linear programming, quadratic programming, simulated annealing, monte-carlo.

Performance Evaluation

Suppose you find that your algorithm's accuracy is 90% and you're very happy about it.

Question: Is the algorithm's performance good?

Answer: ???

You need to define a way to measure your algorithm's performance.

Example: Image Registration

Suppose you know that for each $\mathbf{x}_i \in S$, the correct corresponding point in S' is \mathbf{x}_i^* . This is called the **ground truth**.

But, your actual computed corresponding point is \mathbf{Ax}'_i .

Then, the error of the algorithm can be defined as

$$E = \sum_{i=1}^n \|\mathbf{Ax}'_i - \mathbf{x}_i^*\|. \quad (8)$$

You need to find a way to measure some kind of ground truth.

Concepts of Optimality I

- **Perfect solution:**
Error = 0.
- **Optimal solution:**
Minimize error or minimize (or maximize) objective function.
Example: Minimum of quadratic error function.
- **Statistical optimum:**
Optimal on the average, best solution on the average.
Example: Bayesian classifier.
- **Probabilistic optimum:**
Have probability 1.0 of finding optimal solution.
Example: Some randomized algorithms can find the optimal solution with probability 1.0.
That is, given enough time, they are guaranteed to find the optimal solution.

Concepts of Optimality II

- **Conditional optimum:**

Optimal under certain conditions.

Example: Support Vector Machine can find the global optimum corresponding to a **given** set of predefined parameter values.

The way to evaluate your algorithm's performance is to compare its performance with those of other algorithms, e.g.,

- **an ideal algorithm or an ideal solution:**

This will set the best results.

Your algorithm's performance should not be too far from the best performance.

- **existing well-known algorithms:**

This will tell the difference between the algorithms.

Your algorithm should perform better than the others.

- **baseline algorithm:**

This is typically easy to implement and sets the lowest performance.

Your algorithm should perform much better than the baseline.

Notes:

- Sometimes, it is too time-consuming to implement other existing algorithms for performance evaluation.
- One way is to systematically degrade your algorithm so that it mimics existing algorithms. Then, you can still do a reasonable performance comparison.
- In the worst case, implement the baseline algorithm and compare with the baseline results.
- In some cases, we don't know the ideal solution in general. But, we can generate realistic **synthetic** data with known ideal solution.

Summary

- A good problem definition is precise and it states the problem requirements and objectives.
- Many multimedia analysis problems can be formulated as optimization problems.
- There are many optimization algorithms available. Choose the appropriate one.
- To solve the problem, often need to refine or transform the initial problem definition.
- To do this well, need to know the problem characteristics very well.
- That means, must analyze and understand the problem very well.
- That means, **MUST** write down problem definition!!!