# AMSC 607 / CMSC 764 Advanced Numerical Optimization
## Fall 2001
## UNIT 4: Special Topics
## PART 3: Global Optimization

Dianne P. O'Leary



---

## Global Optimization

**Local Optimization Problem:** Find $x^* \in \mathcal{S}$ such that

$$f(x^*) \leq f(x)$$

for all $x \in \mathcal{S}$ **that satisfy $\|x - x^*\| \leq \epsilon$ for some number $\epsilon > 0$**.

**Global Optimization Problem:** Find $x^* \in \mathcal{S}$ such that

$$f(x^*) \leq f(x)$$

for all $x \in \mathcal{S}$.

Note that we now demand that $x^*$ be the **best** point, not just the **locally best**.

**References:**

- Global Optimization Website
  http://www.mat.univie.ac.at/ neum/glopt.html
- R. Horst and P.M. Pardalos (eds.), *Handbook of Global Optimization*, Kluwer, Dordrecht, 1995.
- There is a journal of *Global Optimization* and there are frequent conferences.

All we have time to do is give a menu of possible approaches and a sample of just a few of them.

---

## Types of methods

---

## Type 1: Heuristic Methods

These methods are not guaranteed to succeed, but on some classes of problems, they work well.

---

### Heuristic Example 1: randomization

Feed **random** starting guesses to your favorite optimization program.

---

### Heuristic Example 2: genetic algorithms

These algorithms carry a set of guesses. At each iteration, each guess is modified (**mutated**) in several ways, and some are chosen to continue, in a **survival of the fittest** strategy.

- These algorithms are easy to program.
- Success depends on adapting the mutations to the problem type.

**Idea:**

- Begin with a set of guesses $\{x^{(k)}\}$, $k = 1, \ldots, N$.
- Form several mutations of each the guesses with probability related to how much the mutation improves the function value.

  (A mutation might be a change of a single coordinate, an interchange of coordinates, a substitution of coordinates from a different guess, etc.)
- Save a subset of the mutants as the next guesses.

---

### Heuristic Example 3: Homotopy algorithms

We embed our problem in a **family of problems** and try to trace the set of local solutions.

Let

$$F(x, t) = tg(x) + (1 - t)c(x)$$

where $g$ is the gradient of our minimization function $f$ and $c$ is the gradient of a convex function with (unique) minimizer $\hat{x}$..

When $t = 0$, there is a unique solution to the equation $F(x, t) = 0$: $x = \hat{x}$.

When $t = 1$, we have our original problem.

So the idea is to increase $t$ gradually from $0$ to $1$, solving the equation $F(x,t) = 0$, for a fixed value of $t$, using Newton's method (for example) started with the solution for the previous value of $t$.

Picture

The branch points, at which new minimizers appear, are signaled by singular Hessian matrices.

---

### Heuristic Example 4: Tunneling

**Reference:** Levy and Montalvo, *SIAM J Scientific and Statistical Computing* 6 (1985) 15-29.

We find a series of better and better local minimizers by getting rid of the ones we have already found.

**Algorithm:** Begin with a guess $\hat{x}$.

1. Find a local minimizer of $f$ using initial guess $\hat{x}$ and your favorite minimization algorithm. The result is $\bar{x}$.
2. Let $f^*$ be the smallest value of $f$ that we have achieved thus far. Set $\{x_i^*\}$ $i = 1, \ldots, \ell$ to be the set of $x$ values that we have already found that achieve the value.
3. Now find a local minimizer of $T$ using initial guess $\bar{x}$ + a random perturbation and your favorite minimization algorithm, where

$$T(x) = \frac{f(x) - f^*}{\prod_{i=1}^{\ell} \|x - x_i^*\|_i^{\eta}}$$

   where $\eta_i$ is chosen "appropriately."
4. Repeat.

---

### Type 2: Transformation methods

- These methods tranform the problem into a more tractable one.
- They only work if the original problem has some known nice properties.

---

### Transformation Example: Branch and Bound

**Example:** If we are trying to minimize subject to a single constraint $0 \le c(x) \le 1$, we can say that the solution to our problem is the better of the solutions to the two problems

$$\min_{0 \le c(x) \le .5} f(x)$$

and

$$\min_{.5 \le c(x) \le 1} f(x) .$$

- For more details, see Grid methods (below).
- Often, the transformation is to a **mixed integer linear programming problem**.

---

### Type 3: Systematic Methods

These methods are guaranteed (under exact arithmetic) to succeed with a predictable amount of work.

---

### Systematic Example 1: simulated annealing

:

This method is provably convergent (with probability 1) but very slow unless enhancements are added.

**Motivation:** Suppose we have a collection of molecules forming a liquid like water. The molecules move somewhat at random:

- When the temperature is high, they have a lot of energy and big movements, even if the movement raises the total energy of the system.

- As the temperature is lower, the energy of each molecule is less, and they tend to move only in directions that decrease the total energy of the system.
- Finally, as we reach the freezing point, the movements are just vibrations within a fixed crystal structure.

The process of lowing the temperature slowly to the freezing point, so that at any time the system is approximately minimizing its energy, is called **annealing**.

**Simulated annealing** (Metropolis, 1952) mimics this process: we add a fictitious temperature parameter $T$ that is slowly reduced. Instead of adjusting the positions of the molecules, we change the entries in $x$, and the energy of the system is measured by $f(x)$.

For a fixed value of $T$, we iterate until $f$ ceases to change much:

- Generate a random change $\Delta x$ for $x$.
- If $f$ is reduced by the change, accept it.
- If $f$ is increased by the change, accept it with probability $e^{(f(x)-f(x+\Delta x))/T}$.

Choosing the parameter $T$, how fast to reduce it, and how long to iterate at each fixed value is an art.

---

### Systematic Example 2: Grid methods

These methods are good if we have some extra information about the function.

**Assume** that $f$ is **Lipschitz continuous**: there exists a constant $c$ such that

$$|f(x) - f(y)| \leq c\|x - y\|$$

for all $x, y \in \mathcal{S}$.

And for ease of illustration, we'll assume that $\mathcal{S}$ is a box.

**Idea:**

- Begin by setting $B = \{\mathcal{S}\}$, $\hat{x} =$ the center of the box $\mathcal{S}$, and $\hat{f} = f(\hat{x})$.
- At each iteration, until the volume of the boxes on the list is small enough,
  - Remove each each box from the list $B$ and subdivide it into 4 boxes.
  - For each of these boxes, find the center point $x_c$ and evaluate $f(x_c)$. If the minimizer could not be in this box

    (i.e., if $f(x_c) > \hat{f} + cr$ where $r$ is the radius of the box),

    then we can drop this box. Otherwise, add this box to the list $B$ for the next iteration.

Picture.

---

**Final words:**

- Global optimization problems are **just plain hard**.
- Heuristics often give the best answers the fastest.
- This is a very **fertile area for research**.