6. SPECIAL METHODS

Consider the box-constrained problem:

minimize $f(\mathbf{x}) = f(x_1,...,x_n)$
subject to $a_j \le x_j \le b_j$, j=1,...,n

6.1. EVOLUTIONARY ALGORITHMS

Sources: Jouni Lampinen's slides at
http://www.it.lut.fi/kurssit/05-06/Ti5216300/Evonet1.pdf
http://www.it.lut.fi/kurssit/05-06/Ti5216300/Evonet2.pdf
contain the basics of evolutionary algorithms and examples.

Evolutionary algorithms mimic the natural evolution mechanisms: reproduction, gene crossover and mutation, survival of the fittest. Evolutionary algorithms can be applied to a wide spectrum of optimization problems with mixed continuous-discrete variables, discontinuous and non-differentiable functions and nonconvex problems.

Examples of evolutionary algorithms:
- *Genetic algorithms*
- *Differential evolution*
- *Evolution strategies*

When to use evolutionary algorithms instead of traditional algorithms?
When nothing can be assumed on continuity, differentiability, unimodality or even if the function cannot be given in excplicit mathematical form. For example when the value of the objective function is a result of experiments, simulation, subjective value or utility.

GENETIC ALGORITHMS

In optimization problems, evolutionary algorithms are based on a *population* of candidate solution vectors. The solutions are represented as strings of binary or floating point numbers called *chromosomes*. Each character in this string corresponds to a *gene*. Let the size of the population be N and the individuals in the current population $\mathbf{x}_1, \mathbf{x}_2,..., \mathbf{x}_N$.

Evolutionary operations
- Parents are selected from the population for *reproduction*. The selection criterion must be specified.
- *Recombination* of genes is carried out by *crossover* of chromosomes of two selected parents. The chromosomes are cut at a random point and the parts are swapped. Alternatively: uniform crossover, where randomly selected genes are swapped between the parents' choromosomes. Recombination produces two descendants (children) of two parents. Each descendant inherits part of its genes from both parents.
- *Mutation* changes individual genes of chromosomes with a specified mutation probability. In a binary string mutation means switching the gene and in a floating point string mutation may be performed with adding random noise to a gene. Mutation is needed for diversity of the gene pool, and it guarantees that the optimum solution can be reached in principle from every starting population (c.f. the global stage of a stochastic global optimization method).
- *Selection*: After reproduction, part of the individuals are selected to form the next generation. This is based on the *fitness* of the individuals. In optimization the measure of fitness is the objective function value. If certain proportion of the best individuals is selected for the next generation and the rest is eliminated, the selection is called *elitistic*.

Example of crossover: Let the individuals be strings of n=5 integers and the crossover site be 2 (i.e. cut after the second digit).

Parents:	Offspring:
$\mathbf{x}_1$ = (2,4,6,1,3)	$\mathbf{y}_1$ = (2,4,1,2,5)
$\mathbf{x}_2$ = (1,7,1,2,5)	$\mathbf{y}_2$ = (1,7,6,1,3)

Uniform crossover, e.g. swap in places 2,3,5:

Offspring: $\mathbf{y}_1$ = (2,7,1,1,3)
$\mathbf{y}_2$ = (1,4,6,2,5)

Outline of a genetic algorithm GA.

1. Generate the initial population of N individuals P_0, k=0.
2. Select parents for reproduction.
3. Perform crossover.
4. Generate mutations.
5. Select the the fittest individuals for the new generation → population P_{k+1}.
6. Terminate if a good enough solution is found or k>kmax. Else, set k=k+1 and go to 2.

Remarks:
• The initial population should be generated using uniform distribution within the box-constraints.
• In selecting individuals for reproduction and mutation it is not always wise to select only the fittest individuals. Allow some randomness: the less fitting individuals may carry useful genetic material. Strategies for selecting parents: fitness proportionate selection, tournament selection.

DIFFERENTIAL EVOLUTION

Differential evolution, as presented in the article

Jouni Lampinen (2001): Global Optimization by Differential Evolution
http://www.it.lut.fi/kurssit/05-06/Ti5216300/DE.pdf

is especially suitable for nonlinear, global optimization problems with continuous variables.

6.2. SIMULATED ANNEALING

The principles of the *simulated annealing* (SA) stem from annealing in metallurgy: the metal is first melted by heating and then cooled so that the configuration of the particles reaches a state of minimum internal energy. The heating phase releases the atoms from their initial positions to wander randomly through states of higher energy. The controlled, slow cooling lets the atoms to find their way to crystalline states of lowest internal energy. If the cooling is too rapid, it causes defects in the crystallization and the reasult is a local minimum of the internal energy.

In optimization each feasible solution vector **x** corresponds to a state (configuration) of the system and f(**x**) corresponds to the energy of the system. The SA method generates a random neighboring solution to the current solution, using some probability distrtibution. If the new solution is better, it is taken as the new iterate. If it is worse, it may still be taken as the new iterate, with some probability depending on the temperature.

The simulated annealing algorithm SA.

1. Choose a starting solution **x** and (a high) initial temperature T.
2. Find a candidate solution **y** by randomly changing the current solution **x**.
   Calculate $\Delta f = f(\mathbf{y}) - f(\mathbf{x})$.
   If $\Delta f \leq 0$, set $\mathbf{x} = \mathbf{y}$.
   If $\Delta f > 0$, set $\mathbf{x} = \mathbf{y}$ with probability $e^{-\Delta f / T}$.
3. If equilibrium is not reached go to 2.
4. If equilibrium with the current temperature is reached, lower the temperature T according to the specified cooling schedule. Repeat from step 2 until convergence is achieved.

Remarks:

• The neighboring solution **y** is generated using some probability distribution, e.g. multinormal distribution centered at **x**.

• The acceptance principle of step 2 is called the *Metropolis criterion* and the acceptance probability is related to *Boltzmann distribution*.

• Equilibrium means that termination criteria for a local optimum are achieved.

• Influence of the temperature T: At the early stages temperature is high and "uphill" moves are allowed. This means "free" wandering around the feasible region. At the later stages the temperature is approaching zero and the probability for "uphill" moves gets smaller. If the cooling is too rapid, the iteration get easily stuck in a local minimum. If the cooling is too slow, the time of computation increases.