Bio-inspired Metaheuristics and Image Processing

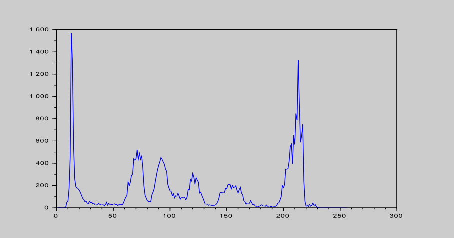
Daniela Zaharie

Department of Computer Science
West University of Timisoara, Romania
e-mail:daniela.zaharie@e-uvt.ro

SSIP 2016

Outline

Multilevel image thresholding

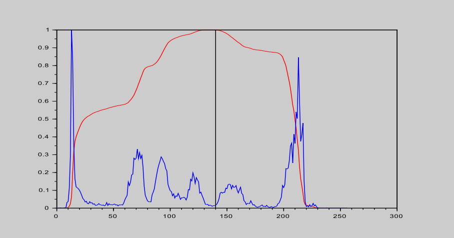


Main idea: find a set of thresholds $0 < t_1 < t_2 < \ldots < t_k < L-1$ s.t. the between class variance, $V(t)$, is maximized

$$V(t_1,\ldots,t_k) = \sum_{i=1}^{k} w_i(\mu_i - \mu)^2 \qquad w_i = \frac{1}{N}\sum_{j=t_{i-1}}^{t_i-1} \mathrm{card}\{(x,y)|I(x,y)=j\}$$

$$\mu_i = \frac{1}{w_i}\sum_{j=t_{i-1}}^{t_i-1} jP_j \qquad \mu = \sum_{j=0}^{L-1} jP_j \qquad P_j = \frac{1}{N}\mathrm{card}\{(x,y)|I(x,y)=j\}$$

Multilevel image thresholding (one threshold)



$k = 1$ - easy to compute

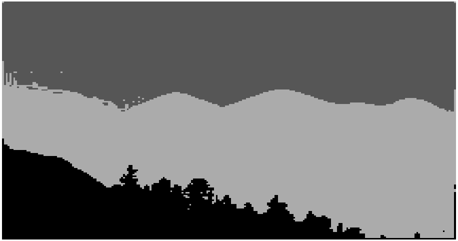
Naive implementation: $\mathcal{O}(\max\{N, L^2\})$

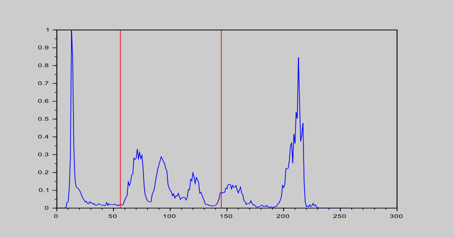
N - image size (number of pixels)

L - number of gray levels

Plot: image histogram (blue), between class variance (red)

Multilevel image thresholding (two thresholds)





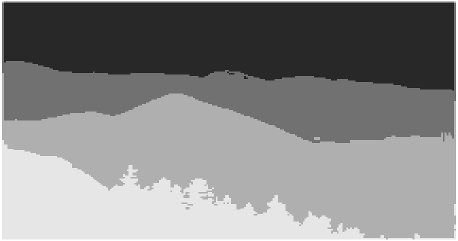
$k = 2$ - easy to compute

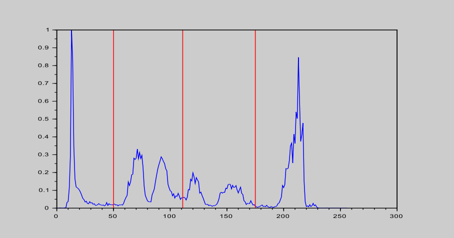
Naive implementation: $\mathcal{O}(\max\{N, L^3\})$

N - image size (number of pixels)

L - number of gray levels

Multilevel image thresholding (three thresholds)





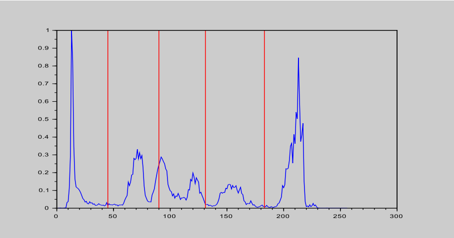
$k = 3$ - easy to compute

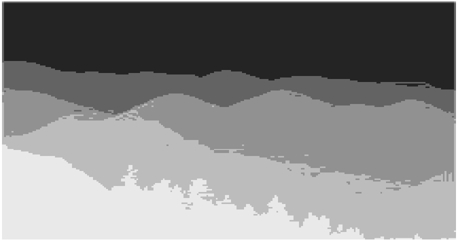
Naive implementation: $\mathcal{O}(\max\{N, L^4\})$

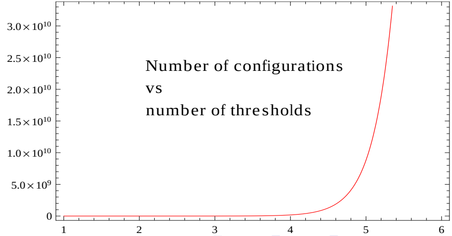
N - image size (number of pixels)

L - number of gray levels

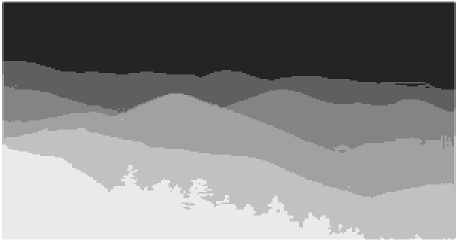
Multilevel image thresholding (four thresholds)

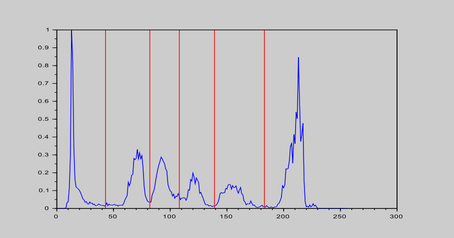






Multilevel image thresholding (five thresholds)





- multilevel thresholding leads to a combinatorial optimization problem
- brute force approaches generate large search spaces; it is not feasible for more than four thresholds
- thresholds obtained by using a population based metaheuristic (Particle Swarm Optimization)

Image segmentation as an optimization problem

- threshold-based: estimate threshold values which maximize variance (Otsu) or entropy (Kapur) measures
  - combinatorial optimization problem with a large search space

Image segmentation as an optimization problem

- threshold-based: estimate threshold values which maximize variance (Otsu) or entropy (Kapur) measures
  - combinatorial optimization problem with a large search space
- cluster-based: estimate centroids in the feature space which optimizes compactness and separation criteria
  - local optimization (e.g. kMeans) methods do not ensure a good exploration of the search space
  - multiple conflicting objectives → multi-objective optimization

Image segmentation as an optimization problem

- threshold-based: estimate threshold values which maximize variance (Otsu) or entropy (Kapur) measures
  - combinatorial optimization problem with a large search space
- cluster-based: estimate centroids in the feature space which optimizes compactness and separation criteria
  - local optimization (e.g. kMeans) methods do not ensure a good exploration of the search space
  - multiple conflicting objectives → multi-objective optimization
- model-based: estimate model parameters which minimize an energy function
  - the energy function might have many optima → global/ multi-modal optimization
  - the parameters should satisfy some constraints → constrained optimization

More on optimization in image processing

non-rigid multi-modal image registration[1]

- find the the parameters of a free form deformation model
- which minimize a similarity measure (e.g. mutual information)
- problem size: for a $8 \times 8 \times 8$ mesh there are 1536 parameters;

[1] Yang at al.,Non-rigid multi-modal medical image registration by combining L-BFGS-B with cat swarm optimization, Information Sciences 2015

[2] Zhang et al., PSO-EM: A Hyperspectral Unmixing Algorithm Based On Normal Compositional Model, TGRS 2014

More on optimization in image processing

non-rigid multi-modal image registration[1]

- find the the parameters of a free form deformation model
- which minimize a similarity measure (e.g. mutual information)
- problem size: for a $8 \times 8 \times 8$ mesh there are 1536 parameters;

(hyper)spectral unmixing[2]

- find the abundancy values which maximizes the log-likelihood function from the E-step in an EM framework
- problem size: abundance map size = number of pixels $\times$ number of endmembers (750 for a 50×50 subimage and 3 endmembers)

$\rightarrow$ large scale optimization

[1] Yang at al.,Non-rigid multi-modal medical image registration by combining L-BFGS-B with cat swarm optimization, Information Sciences 2015

[2] Zhang et al., PSO-EM: A Hyperspectral Unmixing Algorithm Based On Normal Compositional Model, TGRS 2014

More on optimization in image processing

Another example:

- given a parameterized module for image registration encapsulated in a proprietary software
- find the parameters values which:
  - ensure that the registration error is smaller than a given threshold
  - the running time is as small as possible

$\rightarrow$ black box optimization problem

More on optimization in image processing

Another example:

- given a parameterized module for image registration encapsulated in a proprietary software
- find the parameters values which:
  - ensure that the registration error is smaller than a given threshold
  - the running time is as small as possible

$\rightarrow$ black box optimization problem

Characteristics of the black box problems

- only partial/uncertain apriori knowledge on the search landscape
- only objective function values are known, no gradient information

Summary of the problem characteristics

- partial or no knowledge on the search landscape (black box optimization)
- many local optima (global optimization)
- several conflicting optimization criteria (multi-objective optimization)
- constraints on the design variables (constrained optimization)
- many variables to estimate (large scale optimization)

→ gradient-based methods unable to find efficiently the solution

→ metaheuristics

Outline

What is a metaheuristic?

- It is a general-purpose (usually stochastic) procedure designed to solve difficult optimization problems [3]

[3] S. Luke, Essentials of metaheuristics, free online book, 2014

What is a metaheuristic?

- It is a general-purpose (usually stochastic) procedure designed to solve difficult optimization problems [3]
- Main characteristics:
  - it does not require specific knowledge of the problem
  - they are appropriate to solve problems for which the search landscape is not well formalized
  - it mainly rely on two main mechanisms: exploration of the search space and exploitation of the knowledge collected during previous search steps

[3] S. Luke, Essentials of metaheuristics, free online book, 2014

What is a metaheuristic?

- It is a general-purpose (usually stochastic) procedure designed to solve difficult optimization problems [3]
- Main characteristics:
  - it does not require specific knowledge of the problem
  - they are appropriate to solve problems for which the search landscape is not well formalized
  - it mainly rely on two main mechanisms: exploration of the search space and exploitation of the knowledge collected during previous search steps
- Types of metaheuristics:
  - trajectory based - the search process describes a trajectory in the search space; only one "searcher" is used
  - population based - they use a population of "searchers" which cooperate and compete

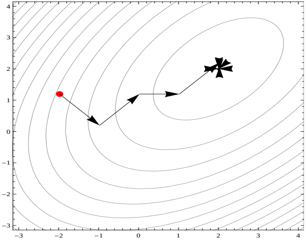
[3] S. Luke, Essentials of metaheuristics, free online book, 2014

Trajectory based metaheuristics

Pattern Search (Hooke-Jeeves local search)[4]

$s \leftarrow s_0$; $r \leftarrow r_0$; $best \leftarrow s$
while $\langle$ NOT stopping $\rangle$ **do**
  $s' \leftarrow s$
  for $j \leftarrow \overline{1,n}$ **do**
    if $f(s \pm r * e_j) < f(s')$ **then**
      $s' \leftarrow s \pm r * e_j$
    end if
  end for
  if $s = s'$ **then**
    $r \leftarrow r/2$ **else** $s \leftarrow s'$
  end if
  if $f(s) < f(best)$ **then**
    $best \leftarrow s$
  end if
end while

Problem to be solved: minimize
$f : [a_1, b_1] \times \ldots \times [a_n, b_n] \rightarrow \mathbb{R}$



- it is a direct search method
- $e_j = (0, .., 0, 1, 0, ...0)$
- $2n$ candidates are analyzed at each step; the best one is kept

[4] T.G. Kolda et al., Optimization by direct search..., SIAM Review, 45(3), 2003

Trajectory based metaheuristics

From local to global optimization:

- Perturbation: use (ocasionally) some large perturbations
- Random restart: new search process started from a random configuration
- Selection: accept (ocasionally) poorer configurations (e.g. Simulated Annealing)
  - inspired by thermodynamics of annealing process
  - objective function interpreted as an energy
  - control parameter T interpreted as temperature
  - from random search ($T \to \infty$) to greedy search ($T \to 0$)

Trajectory based metaheuristics

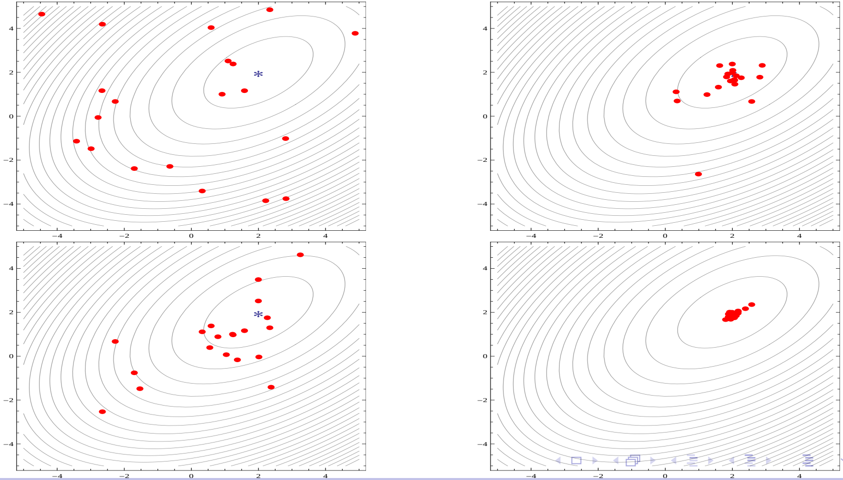
From local to global optimization:

- Perturbation: use (ocasionally) some large perturbations
- Random restart: new search process started from a random configuration
- Selection: accept (ocasionally) poorer configurations (e.g. Simulated Annealing)
  - inspired by thermodynamics of annealing process
  - objective function interpreted as an energy
  - control parameter T interpreted as temperature
  - from random search ($T \to \infty$) to greedy search ($T \to 0$)

```
s ← s_0; t ← 0; T(0) ← T_0
best ← s
while ⟨ NOT stopping ⟩ do
    s' ← perturb(s)
    if rand(0,1) < min{1, exp(-(f(s')-f(s))/T(t))} then
        s ← s'
    end if
    t ← t + 1
    compute T(t) (cooling schedule)
    if f(s) < f(best) then
        best ← s
    end if
end while
return best
```

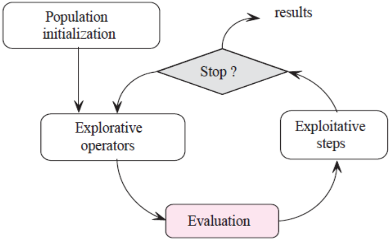
Population based metaheuristics

... a population of searchers (candidate solutions)



Population based metaheuristics: overall structure

1: Initialize a population of m candidates ($P = (x_1, x_2, \ldots, x_m)$)
2: **while** ⟨ Not stopping condition ⟩ **do**
3: Evaluate the population (compute $(f(x_1), f(x_2), \ldots, f(x_m))$)
4: Apply explorative/exploitative mechanisms
5: **end while**



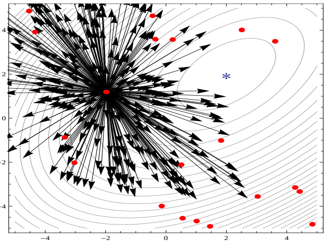
Population Based Iterative Search

Population based metaheuristics: search mechanisms

Overall goal: find the trade-off between

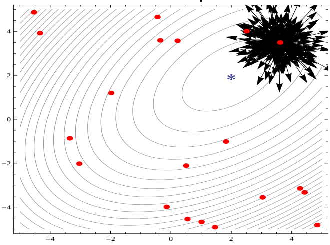
Exploration (diversification) = explore the search space on a global scale

- (large) perturbation of the elements in the current population → discover new promising regions in the search space



Exploitation (intensification) = focus the search in promising regions

- good elements have higher chance to be preserved in the population
- (small) guided perturbation of good elements → local improvement



Population based metaheuristics: design aspects

The design of a metaheuristic is a decision process. Several questions should be answered:

- How should be encoded the population elements?
  - binary vectors, vectors of integer values (e.g. multilevel thresholding), vectors of real values (e.g. registration, deformable models)

Population based metaheuristics: design aspects

The design of a metaheuristic is a decision process. Several questions should be answered:

- How should be encoded the population elements?
  - binary vectors, vectors of integer values (e.g. multilevel thresholding), vectors of real values (e.g. registration, deformable models)
- Which exploration/ exploitation mechanisms should be used?
  - perturbation based on distribution probabilities not related to the population
  - perturbation based on the distribution of the population elements

Population based metaheuristics: design aspects

The design of a metaheuristic is a decision process. Several questions should be answered:

- How should be encoded the population elements?
  - binary vectors, vectors of integer values (e.g. multilevel thresholding), vectors of real values (e.g. registration, deformable models)
- Which exploration/ exploitation mechanisms should be used?
  - perturbation based on distribution probabilities not related to the population
  - perturbation based on the distribution of the population elements
- How much randomness?
  - purely random perturbation or random control parameters may help to preserve/ increase the diversity of the population $\rightarrow$ avoid stagnation

Population based metaheuristics: design aspects

- How many elements in the population?
  - sometimes related to the problem size
  - adaptive population size

Population based metaheuristics: design aspects

- How many elements in the population?
  - sometimes related to the problem size
  - adaptive population size
- When should be stopped the iterative process?
  - pre-specified number of iterations or objective function evaluations
  - until there is no progress

Nature as source of inspiration

Evolution by natural selection

- Main idea: evolution can produce highly optimised processes and structures by
  - reproduction = creation of new elements based in existing ones and on random events
  - selection = survival of the fittest
- Example: evolutionary algorithms (EA)

Nature as source of inspiration

Evolution by natural selection

- Main idea: evolution can produce highly optimised processes and structures by
  - reproduction = creation of new elements based in existing ones and on random events
  - selection = survival of the fittest
- Example: evolutionary algorithms (EA)

Intelligent collective behavior (swarm intelligence)

- Main idea: cooperation between agents following simple rules may lead to complex behavior
- Examples:
  - particle swarm optimization (PSO)
  - ant colony optimization (ACO)
  - artificial bee colony (ABC)
  - firefly algorithm (FA), cuckoo search (CS), flower pollination algorithm (FPA), bat algorithm (BA), bacterial foraging (BFOA) etc.

Nature as source of inspiration

Applications in image processing:

- image enhancement: PSO, ABC, DE
- multi-level thresholding [5]: ACO, PSO, DE, FA, BA, CS
- image registration: DE, BFOA
- segmentation by deformable models[6]: EA, ACO, PSO, CMA-ES, DE

[5] T. Kurban et al, Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding, ASOC 2014

[6] P. Mesejo et al, A survey on image segmentation using metaheuristic ..., ASOC 2016

[7] A. Sorensen, Metaheuristics - the metaphore exposed, ITOR 2013

Nature as source of inspiration

Applications in image processing:

- image enhancement: PSO, ABC, DE
- multi-level thresholding [5]: ACO, PSO, DE, FA, BA, CS
- image registration: DE, BFOA
- segmentation by deformable models[6]: EA, ACO, PSO, CMA-ES, DE

A word of caution[7]: just the fact that a method tries to mimick some behavior which proved to be successful in nature does not ensure the success as an optimization method; in many cases they are just new ways of selling existing ideas - most important is to identify the appropriate search mechanisms

[5] T. Kurban et al, Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding, ASOC 2014

[6] P. Mesejo et al, A survey on image segmentation using metaheuristic ..., ASOC 2016

[7] A. Sorensen, Metaheuristics - the metaphore exposed, ITOR 2013

Population based metaheuristics: Evolutionary Algorithms

- Source of inspiration: evolution of biological species
  - inheritance of the ancestors characteristics $\rightarrow$ crossover (recombination)
  - unexpected influence of the environment and errors in the DNA transcription/ translation processes $\rightarrow$ mutation
  - survival of the fittest $\rightarrow$ selection

Population based metaheuristics: Evolutionary Algorithms

- Source of inspiration: evolution of biological species
  - inheritance of the ancestors characteristics → crossover (recombination)
  - unexpected influence of the environment and errors in the DNA transcription/ translation processes → mutation
  - survival of the fittest → selection
- Encoding
  - vector of binary values → Genetic Algorithms
  - vector of real values → Evolutionary Strategies
  - tree/graph-like structures → Genetic Programming

Population based metaheuristics: Evolutionary Algorithms

Problem to be solved: maximize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

EA building blocks

- Crossover: two or several parents $\to$ one or several offsprings
  - Parents: $(x_1, x_2, \ldots, x_n)$, $(x'_1, x'_2, \ldots, x'_n)$
  - Uniform crossover: $y_i = x_i$ with probability p and $y_i = x'_i$ with probability $1 - p$
  - Arithmetic recombination: $y_i = (x_i + x'_i)/2$

Population based metaheuristics: Evolutionary Algorithms

Problem to be solved: maximize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \rightarrow \mathbb{R}$

EA building blocks

- Crossover: two or several parents $\rightarrow$ one or several offsprings
  - Parents: $(x_1, x_2, \ldots, x_n)$, $(x'_1, x'_2, \ldots, x'_n)$
  - Uniform crossover: $y_i = x_i$ with probability p and $y_i = x'_i$ with probability $1 - p$
  - Arithmetic recombination: $y_i = (x_i + x'_i)/2$
- Mutation: one element and a perturbation model $\rightarrow$ one perturbed element
  - Purely random perturbation: $z_i^j \sim \mathcal{U}(a_j, b_j)$
  - Additive perturbation: $z_i \leftarrow y_i + \xi_i$, ξ_i - random variable (e.g. uniform, normal, Cauchy, Lévy distributions)

Population based metaheuristics: Evolutionary Algorithms

Problem to be solved: maximize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \rightarrow \mathbb{R}$

EA building blocks

- Selection: construct a new generation from the population of parents and/or offsprings
  - Proportional: random sampling with replacement based on a distribution probability related to the elements quality ($Prob(x_i) \approx Fitness(x_i)$) - known as "roulette wheel" procedure
  - Tournament: repeated random sampling of small population subsets + selection of the best element of the subset
  - Truncation: select the best elements out of the joined population (parents and offsprings)

Population based metaheuristics: Evolutionary Algorithms

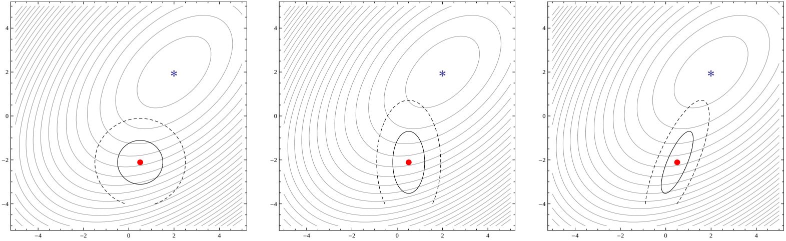
Problem to be solved: maximize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

EA building blocks

- Selection: construct a new generation from the population of parents and/or offsprings
  - Proportional: random sampling with replacement based on a distribution probability related to the elements quality ($Prob(x_i) \approx Fitness(x_i)$) - known as "roulette wheel" procedure
  - Tournament: repeated random sampling of small population subsets + selection of the best element of the subset
  - Truncation: select the best elements out of the joined population (parents and offsprings)
- Elitism: preserve the best element(s) in the new population

Population based metaheuristics: CMA-ES

- mutation based on multi-variate normal distribution with adaptive covariance matrix
- the perturbation used to guide the search exploits the correlation between variables - appropriate in the case of nonseparable problems
- currently the most effective Evolution Strategy [8]



[8] N. Hansen, https://www.lri.fr/~hansen/cmaesintro.html

Population based metaheuristics: CMA-ES

Simplest variant: rank-one update of the covariance matrix

Notations: $x_{i:m}$ element with ith rank (out of the m elements of the population) in decreasing order of quality

`Initialization:` $\mu \in \mathbb{R}$, $C = I$ (identity matrix), $\sigma = 1$, $c = 2/n^2$

`while` ⟨ NOT termination ⟩ `do`

- `Sampling (mutation):` $x_i = \mu + \sigma y_i$, $y_i \sim \mathcal{N}_i(0, C)$, $\quad i = \overline{1, m}$
- `Update mean (selection+recombination):`

$$\mu_{new} = \sum_{i=1}^{k} w_i x_{i:m} = \mu_{old} + \sigma \sum_{i=1}^{k} w_i y_{i:m}, (w_1 \geq w_2 \ldots \geq w_k, \sum_{i=1}^{k} w_i = 1)$$

- `Update covariance matrix (selection+recombination):`

$$C = (1-c)C + c\lambda_w y_w y_w^T, \quad \lambda_w = \frac{1}{\sum_{i=1}^{k} w_i^2}, \quad y_w = \sigma \sum_{i=1}^{k} w_i y_{i:m}$$

Population based metaheuristics: CMA-ES

Improved variant (for large populations): rank-m update of the covariance matrix

Initialization: $\mu \in \mathbb{R}$, $C = I$ (identity matrix), $\sigma = 1$, $c = k/n^2$
while ⟨ NOT termination ⟩ do

- Sampling (mutation): $x_i = \mu + \sigma y_i,\ y_i \sim \mathcal{N}_i(0, C), \quad i = \overline{1, m}$
- Update mean (selection+recombination):

$$\mu_{new} = \sum_{i=1}^{k} w_i x_{i:m} = \mu_{old} + \sigma \sum_{i=1}^{k} w_i y_{i:m}, (w_1 \geq w_2 \ldots \geq w_k, \sum_{i=1}^{k} w_i = 1)$$

- Update covariance matrix (selection+recombination):

$$C = (1 - c)C + c \sum_{i=1}^{k} w_i y_{i:m} y_{i:m}^T$$

Population based metaheuristics: CMA-ES

Full version: cumulation-based update of C + adaptive *sigma*

`Initialization:` $\mu \in \mathbb{R}, \sigma \in \mathbb{R}_+$, $C = I, p_C = 0, p_\sigma = 0$, $c_C = c_\sigma = 4/n$, $c_1 = 2/n^2$, $c_m = 0.3m/n^2$ (s.t. $c_1 + c_m \leq 1$, $d_\sigma = 1 + \sqrt{m_w/n}$, $m_w = 0.3m$

`while` ⟨ NOT termination ⟩ `do`

- `Sampling:` $x_i = \mu + \sigma y_i$, $y_i \sim \mathcal{N}_i(0, C)$, $\quad i = \overline{1, m}$
- `Update mean:` $\mu_{new} = \sum_{i=1}^{k} w_i x_{i:m} = \mu_{old} + \sigma \sum_{i=1}^{k} w_i y_{i:m}$,
- `Cumulation for` C `and` σ `update:`
  $p_C = (1 - c_C)p_C + \mathbf{1}_{\|p_\sigma\| < 1.5\sqrt{n}} \sqrt{1 - (1 - c_C)^2} \sqrt{m_w} y_w$
  $p_\sigma = (1 - c_\sigma)p_\sigma + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{m_w} C^{-1/2} y_w$
- `Update` C `and` σ`:`
  $C = (1 - c_1 - c_m)C + c_1 b_c p_C^T + c_m \sum_{i=1}^{k} w_i y_{i:m} y_{i:m}^T$
  $\sigma = \sigma \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|p_\sigma\|}{E\|\mathcal{N}(0,I)\|} - 1\right)\right)$

Remark: CMA-ES source code: `https://www.lri.fr/~hansen/cmaesintro.html`

Population based metaheuristics: CMA-ES

Successful applications in image processing:

- registration of intraoperative 3D ultrasound data of the spine with preoperative CT data[9]:
- multiple targets detection in image sequences [10] $\rightarrow$ fit a parameterized model (e.g. an ellipse) to the image data

[9] S. Winter et al. Registration of bone structures ..., CARS 2005

[10] J. Brunger et al. Randomized global optimization for robust pose estimation of multiple targets in image sequences, Mathematical Models and Computational Methods, 2015

Particle Swarm Optimization

Source of inspiration: birds flocking, fish schooling[11]

Initialization:

- particles position: $x_i^j = U(a_j, b_j), \quad i = \overline{1,m}, j = \overline{1,n}$
- velocity: $v_i = 0, \quad i = \overline{1,m}$
- global and personal best position: $gbest$, $(pbest_1, \ldots, pbest_m)$

while $\langle$ NOT termination $\rangle$ do

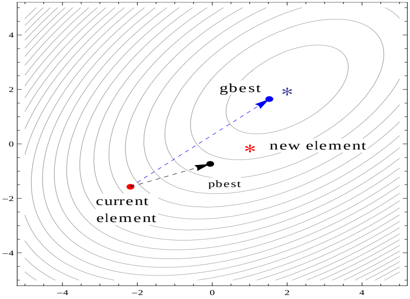
- Evaluation: compute $(f(x_1), \ldots, f(x_m))$;
  update $(gbest)$ and $(pbest_1, \ldots pbest_m)$
- Velocity update:

$$v_i \leftarrow \gamma v_i + \underbrace{c_1 \cdot rand(0,1) \cdot (gbest - x_i)}_{\text{social term}} + \underbrace{c_2 \cdot rand(0,1) \cdot (pbest_i - x_i)}_{\text{cognitive term}}, \quad i = \overline{1,m}$$

- Position update: $x_i \leftarrow x_i + v_i, \quad i = \overline{1,m}$

[11] J. Kennedy, R. Eberhart, Particle Swarm Optimization, 1995

Particle Swarm Optimization

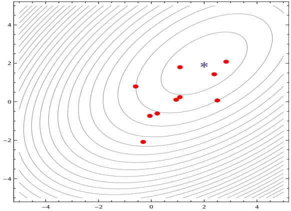


Remarks:

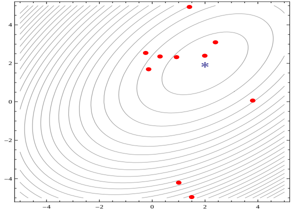
- $\gamma \in (0,1)$, $c_1 \in (0,4)$, $c_2 \in (0,4)$ are control parameters
- use a directed mutation (guided by the best element in the swarm and best experience of each agent)
- no selection
- randomness ensured only by the coefficients of the difference-based terms

Particle Swarm Optimization

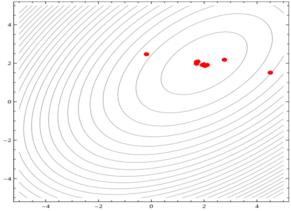
$gen = 0$



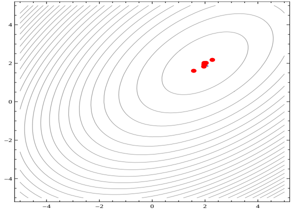
$gen = 10$



$gen = 20$



$gen = 50$



Ant Colony Optimization/ Systems

- Source of inspiration:
  - behaviour of ants when searching for food
  - indirect communication between ants through pheromone trails (stigmergy)
- Main ideas:
  - describe the problem as search on a graph
  - several communicating agents (ants) are used to explore the search space and construct solutions in an incremental way
  - each agent visits several nodes in the graph using a probabilistic decision rule
  - good trajectories are enhanced

Ant Colony Optimization/ Systems

Decision rule - transition probability from node i to node j for ant k

$$P_{ij}^{k} = \begin{cases} \frac{\tau_{ij}^{\alpha} \eta_{ij}^{\beta}}{\sum_{l \in N(k,i)} \tau_{il}^{\alpha} \eta_{il}^{\beta}} & j \in N(k,i), \text{not visited} \\ 0 & \text{otherwise} \end{cases}$$

τ_{ij} - relevance of edge (i,j) based on previous experience (amount of pheromone)
η_{ij} - relevance of edge (i,j) based on prior knowledge (heuristic incorporating particularities of the problem)

Ant Colony Optimization/ Systems

Decision rule - transition probability from node i to node j for ant k

$$P_{ij}^{k} = \begin{cases} \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{l\in N(k,i)} \tau_{il}^{\alpha}\eta_{il}^{\beta}} & j \in N(k,i), \text{not visited} \\ 0 & \text{otherwise} \end{cases}$$

τ_{ij} - relevance of edge (i,j) based on previous experience (amount of pheromone)

η_{ij} - relevance of edge (i,j) based on prior knowledge (heuristic incorporating particularities of the problem)

Pheromone updating rule: $\tau_{ij}^{new} = \underbrace{(1-\rho)\tau_{ij}^{old}}_{\text{evaporation}} + \underbrace{\rho\sum_{k=1}^{m}\Delta\tau_{ij}^{k}}_{\text{deposition}}$

τ_{ij}^{k} - contribution of ant k to the visited edge (i,j)

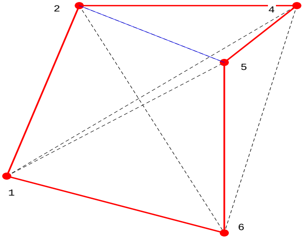
- Evaporation: help forget previous bad decisions
- Deposition: reinforce the influence of good decisions

Ant Colony Optimization/ Systems

Overall structure

Initialize pheromone matrix (*tau*)
Initialize ants positions
for $g \leftarrow 1, genMax$ **do**
    for $s \leftarrow 1, stepMax$ **do**
        for $k \leftarrow 1, m$ **do**
            move ant k
            pheromone update (local)
        end for
    end for
    evaluate the configuration
    pheromone update (global)
end for

Typical application: Travelling Salesman Problem

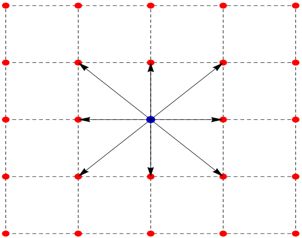


- each ant constructs a route
- $\eta_{ij} = 1/d(i,j)$
- $\Delta\tau_{ij}^{k}$ related to the quality of the tour constructed by ant k

Ant Colony Optimization/ Systems

Initialize pheromone matrix (tau)
Initialize ants positions
for $g \leftarrow 1, genMax$ **do**
    for $s \leftarrow 1, stepMax$ **do**
        for $k \leftarrow 1, m$ **do**
            move ant k
            pheromone update (local)
        end for
    end for
    pheromone update (global)
end for

Edge detection



pheromone matrix (or visit matrix) $\rightarrow$ output image

- both the heuristic and pheromone information is associated to a node
- $\eta_{(x,y)} = |I(x-1, y-1) - I(x+1, y+1)| + |I(x-1, y+1) - I(x+1, y-1)| + |I(x-1, y) - I(x+1, y)| + |I(x, y-1) - I(x, y+1)|$
- the new position is searched amongst the neighbours in the image grid

Differential Evolution

Problem to be solved: minimize $f : [a_1, b_1] \times \ldots \times [a_n, b_n] \to \mathbb{R}$

Main particularity: difference-based mutation[12]

Initialization: $x_i^j = U(a_j, b_j), \quad i = \overline{1, m}, j = \overline{1, n}$

`while` $\langle$ NOT termination $\rangle$ `do`

- Mutation:

$$y_i = x_{r_1} + F \cdot (x_{r_2} - x_{r_3}), \quad i = \overline{1, m}$$

- Crossover:

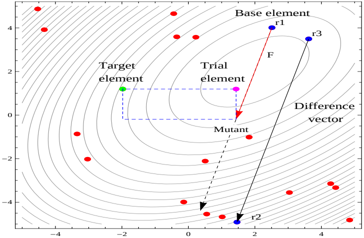
$$z_i^j = \begin{cases} y_i^j & \text{if } rand(0,1) < CR \text{ or } j = j_0 \\ x_i^j & \text{otherwise} \end{cases}, \quad i = \overline{1, m}, j = \overline{1, n}$$

- Selection:

$$x_i(g+1) = \begin{cases} z_i & \text{if } f(z_i) \leq f(x_i(g)) \\ x_i & \text{if } f(z_i) > f(x_i(g)) \end{cases}, \quad i = \overline{1, m}$$

[12] R. Storn, K. Price, Differential Evolution, 1995

Differential Evolution



Decisions at design:

- mutation and crossover variants - various explorative/exploitative abilities
- population size
- control parameters: scale factor (F), crossover parameters (CR)

Remark: there exist various adaptive and (self)adaptive variants (e.g. jDE, JADE, SaDE, SHADE etc)

Differential Evolution

DE taxonomy: $\boxed{\text{DE/ base element/ no. of differences/ crossover type}}$

- Base element:
  - random(x_{r_1}): DE/rand/*/*
  - best (x_*): DE/best/*/*
  - combination of current and best elements ($\lambda x_* + (1-\lambda)x_i$): DE/current-to-best/*/*
  - combination of random and best elements ($\lambda x_* + (1-\lambda)x_{r_1}$): DE/rand-to-best/*/*
  - combination of current and random elements ($\lambda x_i + (1-\lambda)x_{r_1}$): DE/current-to-rand/*/*
- Number of differences: usually 1 (DE/*/1/*) or 2 (DE/*/2/*)
- Crossover type: binomial: DE/*/*/bin, exponential: DE/*/*/exp

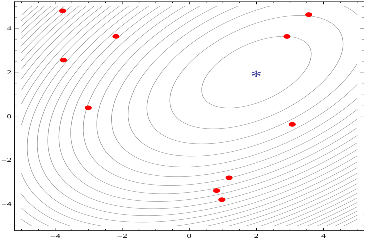
At least 20 DE variants ... currently the most effective are JADE, L-SHADE

Outline

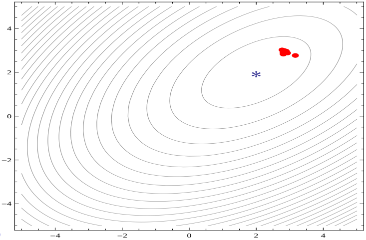
... DE case

Example:

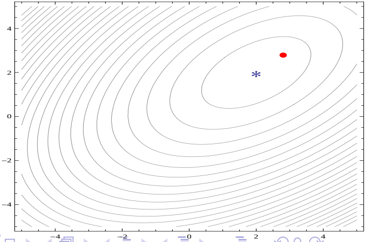
- DE/rand/1/bin, Neumaier function, $n = 2$
- $m = 10$, $CR = 0.9$, $F = 0.2$



$gen = 1$



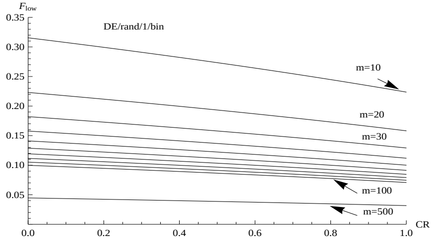
$gen = 5$



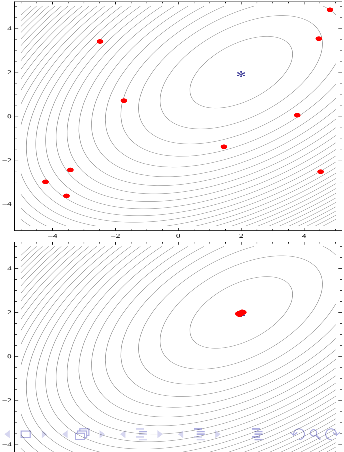
$gen = 15$

... DE case

- Main cause: loss of population diversity
- Solutions:
  - increase randomness
  - control the amount of perturbation (e.g. $F > F_{low}$)
  - multiple subpopulations with limited communication between them



$F = 0.5$ $(m = 10,$
$CR = 0.9 \rightarrow F_{low} = 0.23)$



How to deal with multiple optimization criteria?

Motivation:

- the energy to be minimized in the case of segmentation based on deformable models contain several terms (e.g. $E(u) = \alpha E_{int}(u) + \beta E_{ext}(u)$) $\rightarrow$ the appropriate weights depend on the image $\rightarrow$ need for tuning [14]
- clustering-based segmentation requires minimization of intra-cluster variance and maximization of inter-cluster variance (or optimization of connectedness and compactness)[15]

[14] J. Novo et al., Evolutionary multiobjective optimization for TANs, Patt Rec 2010

[15] C. Bong, Multi-objective nature-inspired clustering and classification techniques for image segmentation, ASOC 2011

How to deal with multiple optimization criteria?

Approach:

- if the components of the energy are conflicting then the problem can be formulated as a multiobjective optimization one
- multiobjective optimization = find the nondominated vectors (Pareto optimal set)
- concept of domination: $x \succ y$ (x dominates y with respect to r criteria $f_1,\ldots,f_r$) if $f_k(x) \leq f_k(y)$ for all $k = \overline{1,r}$ and there exists at least one $j \in \{1,\ldots,k\}$ s.t. $f_j(x) < f_j(y)$

How to deal with multiple optimization criteria?

Which element is better?

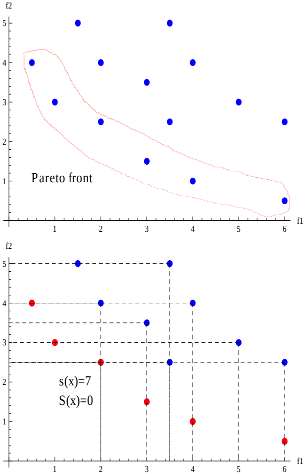
- Strength-based score (SPEA*)
  - Strength of an element

$$S(x) = \sum_{y \in Dom(x)} s(y)$$

$$s(y) = \mathrm{card}(Dom(y))$$

$$Dom(x) = \{y | y \succ x\}$$

  - smaller $S(x)$ means better x

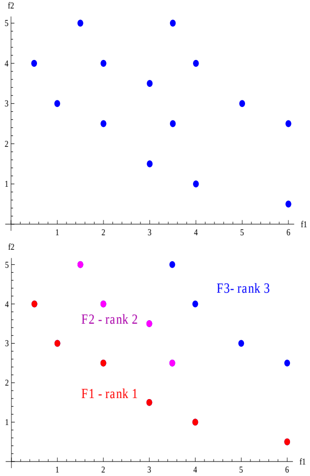


[SPEA = Strength Pareto Evolutionary Algorithm (Zitzler, 2001)]

How to deal with multiple optimization criteria?

Which element is better?

- Nondomination rank (NSGA*)
  - rank 1: elements of F_1 (nondominated elements of a population P)
  - rank 2: elements of F_2 (nondominated elements of $P \setminus F_1$
  - rank 3: elements of F_3 (nondominated elements of $P \setminus (F_1 \cup F_2)$
  - ...
  - elements selected in increasing order of the ranks



[NSGA = Nodominated Sorting Genetic Algorithm (K. Deb, 2002)]

How to deal with multiple optimization criteria?

Particularities of metaheuristics for multi-objective optimization

- Archiving: non-dominated elements are preserved in an archive
  - once a new element is added all those who are dominated are removed
  - the archive is periodically reorganized in order to keep its size under a given upper bound
- Diversity preserving: the approximation of the Pareto set should be diverse and it should "cover" in a uniform way the true Pareto set
  - crowding selection criteria: elements in less crowded regions are preferred

How to deal with constraints?

Bounding box constraints: $x \in [a, b]$

Repairing rules:

- iterate the reproduction operator until the offspring satisfies the constraint
- use a mirroring rule, i.e. when $x \notin [a, b]$ iterate:

$$x' = \begin{cases} b - (x - b) & \text{if } x > b \\ a + (a - x) & \text{if } x < a \end{cases}$$

  until $x' \in [a, b]$.

- select randomly an element in the search range

How to deal with constraints?

Arbitrary constraints

Find x which minimizes $f(x)$ subject to

- $g_i(x) \leq 0, \quad i = 1, \ldots, p$
- $h_j(x) = 0, \quad j = 1, \ldots, q$ (usually transformed in $|h_j(x)| < \epsilon$)

Penalty functions

- minimize

$$\alpha f(x) + \beta \sum_{i=1}^{p} G(g_i(x)) + \gamma \sum_{j=1}^{q} |h_j(x)|$$

$$G(u) = \begin{cases} 0 & u \leq 0 \\ u & u > 0 \end{cases}$$

- Advantage: only the objective function is changed
- Disadvantage: it requires parameters

How to deal with constraints?

Arbitrary constraints - feasibility rules

Deb's feasibility rule: use separate objective value (f) and penalty value (degree of constraint violation - ϕ) when compare two elements; x is better than x' if:

- x and x' are both feasible and $f(x) < f(x')$
- x is feasible and x' is not feasible
- x and x' are both unfeasible and $\phi(x) < \phi(x')$

Advantages:

- easy to implement and to combine with various search algorithms
- it does not require parameters

Disadvantages:

- separating the constraints and the objective function can lead to diversity loss (because they strongly favour the feasible solutions)

How to deal with constraints?

Arbitrary constraints - stochastic ranking

Main idea of stochastic ranking: decides randomly which selection criterion to use (objective or penalty function) x is better than x' if

$$\begin{cases} ((\phi(x) = \phi(x') = 0) \text{ or } (\text{rand}(0,1) < P_f)) \text{ and } (f(x) < f(x'))) \\ \phi(x) < \phi(x') \end{cases}$$

- Advantage: it limits the diversity loss (by accepting promising but unfeasible candidates)
- Disadvantage: it requires the specification of a parameter (P_f, e.g. $P_f = 0.45$)

How to deal with large scale problems?

Cooperative coevolution: split the problem into smaller sub-problems

- a potential solution consists of several components
- evolve independently the population corresponding to each component (coevolution)

  $\rightarrow$ assignment of the variables to components (correlated variables should be in the same component)[16]
- each component is evaluated in the context of other components (cooperation)

  $\rightarrow$ decision on the context selection (random vs. elitistic)

[16] M. Omidvar et al., Cooperative co-evolution with delta grouping for large scale for non-separable function optimization, CEC 2010

How to deal with large scale problems?

Parallelization models

- Objective function evaluation ⇒ master-slave model
    - the master process executes the iterative process
    - the slaves only evaluate the population elements (e.g. simulation of a process, costly computations)

How to deal with large scale problems?

Parallelization models

- Objective function evaluation ⇒ master-slave model
  - the master process executes the iterative process
  - the slaves only evaluate the population elements (e.g. simulation of a process, costly computations)
- Large size population ⇒ island model
  - the population is divided into several subpopulations on which the same or different algorithms are executed
  - the subpopulations communicate by transferring elements according to a given topology

How to deal with large scale problems?

Parallelization models

- Objective function evaluation ⇒ master-slave model
  - the master process executes the iterative process
  - the slaves only evaluate the population elements (e.g. simulation of a process, costly computations)
- Large size population ⇒ island model
  - the population is divided into several subpopulations on which the same or different algorithms are executed
  - the subpopulations communicate by transferring elements according to a given topology
- Costly reproduction operators ⇒ cellular model
  - each population element is assiged to a processor
  - the reproduction and selection are defined at a neighborhood level

Outline

Summary: Pros and Cons

Pros:

- general purpose methods
- almost no requirements on the objective functions
- able to deal with multi-modal functions
- easy to be implemented
- implicit parallelism

Cons:

- mainly based on empirical validation
- limited theoretical results - no guarantees concerning the behavior
- design based on (sometimes many) user decisions

Summary: Selection criteria

- appropriateness - ability to deal with the problem characteristics
- competitiveness - good behavior for similar problems
- simplicity - easy to understand/ implement
- availability - easy to find implementations

Summary: Selection criteria

- appropriateness - ability to deal with the problem characteristics
- competitiveness - good behavior for similar problems
- simplicity - easy to understand/ implement
- availability - easy to find implementations

Take home message: use bio-inspired metaheuristics only when traditional methods do not work or when there is limited knowledge on the particularities of the optimization problem