TOPICS IN EVOLUTIONARY COMPUTATION AND MODEL-BASED CLUSTERING

Sharon McNicholas

Talk Overview

- Topics in evolutionary computation and mixture model-based clustering are explored, as well as the intersection of the two.
- First, complex fitness landscapes and evolutionary computation techniques are explored using cellular automata.
- Next, (Gaussian) mixture model-based approaches to clustering are considered, and a parameter estimation approach using evolutionary computation is proposed and illustrated.

Outline

EA Background

- A computer algorithm incorporates some of the elements of the biological theory of evolution.
- Evolutionary operations are performed on members of a population, which reproduce and create new population members.
- The new members replace less "fit" members from the previous generation, and the process is continued until some stopping criterion is met.
- Evolutionary operations include actions such as crossover and mutation.
- The measure of "fitness" used is determined by the goal of the evolution, i.e., what parameter or process is being optimized.

Terminology

- *Variation operators* are used to produce variation in population members, e.g., crossover and mutation operators.
- *Crossover:* The combining of two data structures to produce at least one new structure. A crossover operator is a binary variation operator, as it acts on two individual population members per execution.
- *Mutation* is the process by which random changes are made to a population member's structure. Mutation can be used to produce a constant supply of minor variation in the population over time. A mutation operator is a unary variation operator, as it acts on one individual population member per execution. The *mutation rate* is the number of individual mutations per structure.

Terminology

- In *fitness based reproduction*, solutions that are deemed to be fitter, based on a predetermined *fitness function*, are preferentially selected to reproduce.
- A *fitness landscape* is a graphical representation of the fitness function. Maxima in the landscape correspond to regions of high fitness.
- *Elitism* occurs when the EA is designed so that a certain number of the most fit individuals are guaranteed to survive and reproduce. This is in comparison to other approaches that incorporate a certain amount of randomness in the selection process.

EA Design Factors

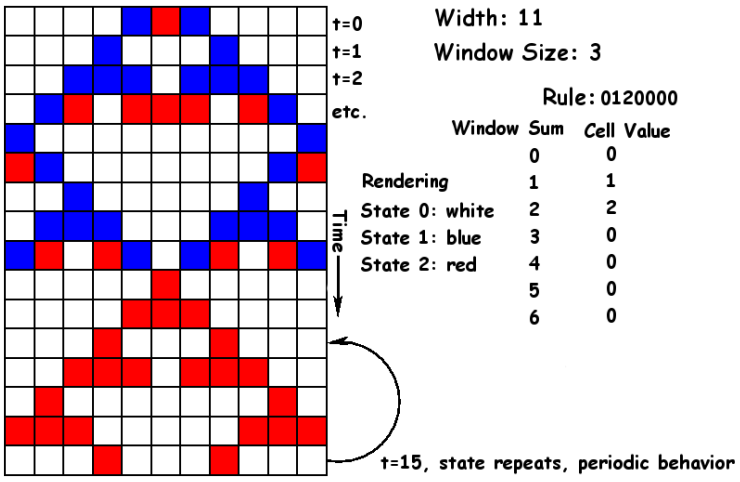
- *Data structure*, e.g. strings or arrays.
- The type of *fitness function*.
- Choice of *variation operator*.
- *Selection rules* for reproduction.
- The *stopping criterion*.

Introduction

Cellular automata are a type of discrete model of computation. A cellular automaton has three parts:

1. A collection of cells. For each individual cell, the surrounding cells that influence the iteration of that cell are called its neighbourhood.
2. A set of possible cell states.
3. A rule that maps the set of possible cell states of a neighbourhood to a new state for the cell with which the neighbourhood is associated.

CA Time History



The first 16 time steps of a synchronous one-dimensional cellular automata, with its rule in two forms.

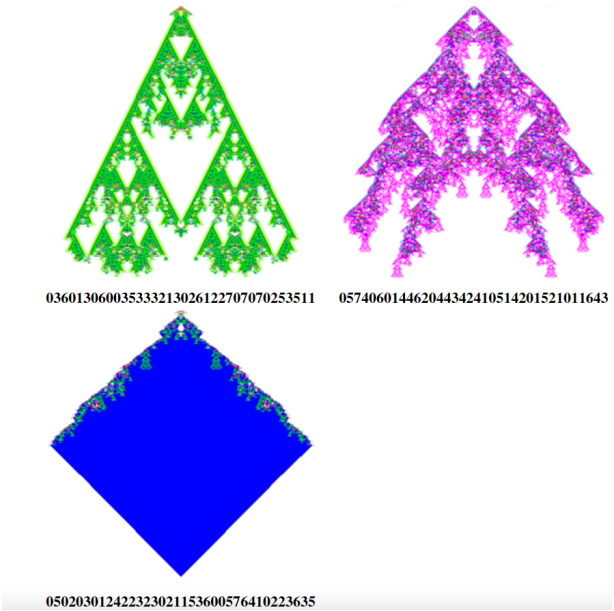
Outline

- This study examines the fitness landscape that arises when evolving synchronous one-dimensional automata to maximize the apoptotic fitness function.
- Apoptotic automata are evolved to fill as much space as possible while dying no later than a pre-specified time.
- The automata in this study used eight cell states and an updating window of size five.
- To update the automata to the next time step 5 cell states are considered; the state of the cell being updated and its two neighbours on each side. These form the window or "neighbourhood" for updating.

Outline II

- The values in a cell's neighbourhood are summed. That sum is then used as an index for a table called the automata's "rule".
- The cell state 0 is thought of as dead while all nonzero states are thought of as living.
- If no live cells remain in the final step then the fitness is the number of live cells in the time history from time zero to the next to last step.
- Eight states were chosen because, in addition to yielding a very large space of rules, eight cell states permit a mapping onto basic RGB colors (black, red, green, blue, magenta, cyan, yellow, and white).

Example Apoptotic CAs



Outline

- The target of evolution in this study are cellular automata updating rules.
- They are represented as arrays of 36 values that must be cell states.
- The variation operators used are two point crossover and m-point mutation in which m values within the rule are changed.
- In another set of experiments, discussed later, an additional crossover operator called "single parent" crossover is used.
- Given that the first cell of the array is forced to be zero, there are 8^{35} rules in the search space.

Selection

- The population is shuffled into groups of four CA-rules.
- The two fitter rules are copied over the two less fit with ties broken uniformly at random.
- The copies are subjected to both crossover and the mutation operator specified by the study.
- An initial 13 experiments yielded 390 best-of-run rules for the apoptotic fitness function. We call this the "standard apoptotic data set" (SADS).

Outline

- Single parent crossover uses a fixed set of "ancestor" rules which population members cross over with.
- The choice of ancestor rules steers the direction of evolutionary search.
- An interesting question was, will the algorithm simply reproduce, after a series of fortuitous single parent crossovers, one of the ancestors or a rule with the same behaviour as one of the ancestors?

Max Fitness

- The mean apoptotic fitness over a large sample of random chromosomes is less than 0.1.
- The maximum fitness located is in the tens of thousands.
- This means that almost the entire fitness landscape is flat: fitness zero. This low mode fitness follows from the fact that most rules never die and so receive an apoptotic fitness of zero.
- A well designed evolutionary algorithm is capable of locating the non-flat portion of the fitness landscape. The analysis of the fitness landscape discounts the majority zero-fitness areas and concentrates on the higher fitness portions which we will demonstrate are quite rugose.
- Rugose fitness landscapes have fitness that jumps up and down along most paths through the fitness landscape. In particular, rugose landscapes are typically rich in local optima.

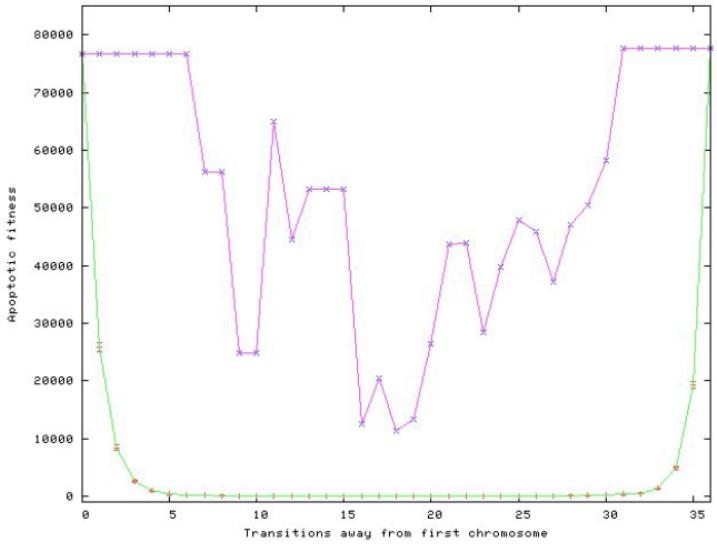
Initial SADs Study

- The 390 experiments in the SADS located 390 distinct optima.
- The minimum fitness found in the SADS is 18,349 while the maximum is 78,558, a ratio a little over 4.25.
- The landscape was very rugose. There was little response to increasing the number of mutations, which suggests that the landscape is not at all smoothed by the increase from one to three point mutations.

Fitness Webs

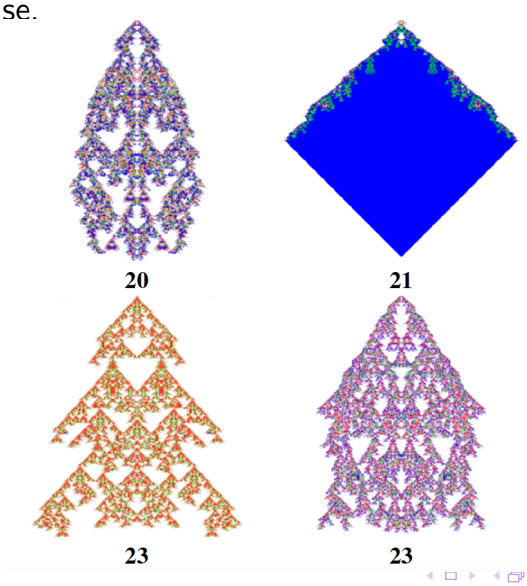
- Start with the first chromosome.
- Computing the fitness at each step, change the value of the chromosome at each successive position in the random order to that of the second chromosome.
- A sample represents one path, stepping by single point mutations, from the first chromosome to the second.
- A set of 10,000 samples is averaged to provide a sense of the space between two chromosomes.

Fitness Web



SADs

Time histories of the 4 automata from the SADS data set that use the smallest number of "active positions". The numbers below are the number of positions they use.



SADs cont.

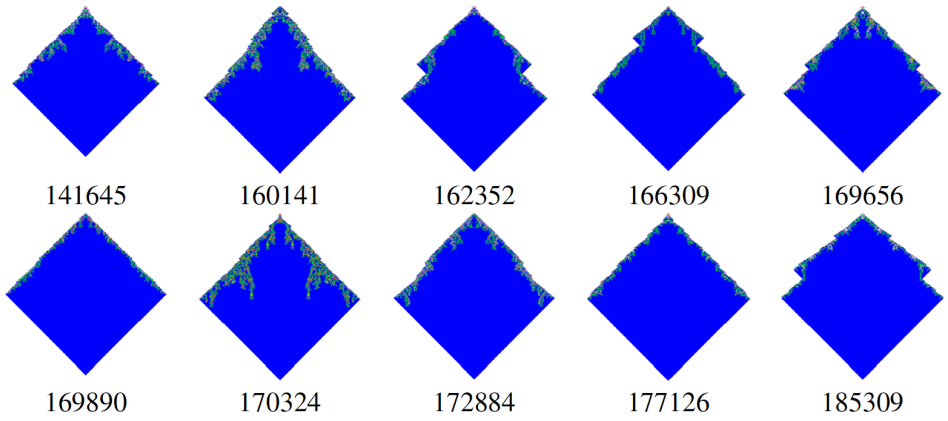
- The number below the CA shows the number of positions in the rule string that are active for an updating event, which is then used for fitness evaluation.
- Any position not used represents a position in the chromosome that can be mutated without changing the time history of the automata, or its fitness.
- The alteration of any of the used positions in the chromosome will result in a changed time history.
- Note that a rule that uses all 36 positions is thus a single-point optimum, while a rule that uses 20 of its 36 positions has a neutral network with at least $8^{36-20} = 8^{16}$ chromosomes.

Outline

- The 390 chromosomes in the SADS data set were used as the ancestors in a single parent experiment.
- The single parent techniques yield 2.21-fold improvement in fitness.
- The single parent technique successfully generalizes the results for one fitness function to another.

Results

10 time histories from the single-parent experiments using chromosomes evolved for the width 401 apoptotic fitness function as ancestors to solve the width 601 apoptotic function. The numbers below are fitness values.



Synopsis

- The apoptotic rules are clustered together with the space "between" high fitness rules in the SADS dense with other high fitness rules.
- The sense of "between" used here is that of the paths constructed for fitness webs.
- The commonness of zero fitness rules in the space, together with the clustering of the apoptotic rules, means that it is likely that the apoptotic rules are dense in a restricted region of the space.
- Single parent techniques yield a substantial improvement in fitness.

Synopsis II

- The success of single parent techniques, for the problem of locating apoptotic automata, exploits the distribution of apoptotic rules within the space.
- Single parent techniques, by continuously re-introducing material from the ancestor set, have the effect of localizing search near the ancestor set.
- Changing the fitness function from a drawing arena 401 across to one 601 across means that we are searching "near" the ancestors on a different fitness landscape.

Applications

- The CA rules can be thought of as self-delimiting pictures; the algorithm does not need to know the size of the picture, it just needs to draw until apoptotic death occurs.
- Single parent evolution of apoptotic cellular automata from a database of rules provides a source of images for on-demand elements for photo-mosaics.
- The system is capable of generalizing solutions from easy cases of the problem to harder cases - reduced evaluation time.

Additional Study

- An extensive study of Single Parent generalization of cellular automata rules was under taken.
- The results are too numerous to discuss here, and can be found in Ashlock, D. and McNicholas, S. (2012) "Single parent generalization of cellular automata rules". IEEE Congress on Evolutionary Computation 1–8.

Classification Terminology

- Classification is a mechanism by which group membership labels are assigned to unlabelled observations.
- The group itself may be in the form of a class or cluster.
- The term class implies that some of the observations are *a priori* labelled; while a cluster is a group of points when all points are *a priori* unlabelled.

Finite Mixture Models

- Finite Gaussian mixture models have model density of the form

$$f(\mathbf{x}) = \sum_{g=1}^{G} \pi_g \phi(\mathbf{x} \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g).$$

  - $\pi_g > 0$ is the probability that an observation belongs to component g ($\sum_{g=1}^{G} \pi_g = 1$), and
  - $\phi(\mathbf{x} \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)$ is the density of a multivariate Gaussian distribution with mean $\boldsymbol{\mu}_g$ and covariance matrix $\boldsymbol{\Sigma}_g$.

- The Gaussian mixture model is the most popular approach within the literature.

Model-Based Clustering: Likelihood

- The Gaussian model-based clustering likelihood for $\mathbf{x}_1, \ldots, \mathbf{x}_n$ is

$$\mathcal{L}(\boldsymbol{\vartheta} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{i=1}^{n} \sum_{g=1}^{G} \pi_g \phi(\mathbf{x}_i \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g),$$

  where $\boldsymbol{\vartheta} = (\pi_1, \ldots, \pi_G, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_G, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_G)$ denotes the model parameters.

- Use z_{ig} to denote component membership, such that $z_{ig} = 1$ if observation i belongs to component g and $z_{ig} = 0$ otherwise.

- Parameter estimation is sometimes carried out based on the complete-data likelihood, i.e.,

$$\mathcal{L}_c(\boldsymbol{\vartheta} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{z}_1, \ldots, \mathbf{z}_n) = \prod_{i=1}^{n} \prod_{g=1}^{G} \left[\pi_g \phi(\mathbf{x}_i \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)\right]^{z_{ig}}.$$

Model-Based Clustering: Predicted Classifications

- The predicted (soft) classifications are the expected values of the Z_{ig}, given by

$$\mathbb{E}[Z_{ig} \mid \hat{\boldsymbol{\vartheta}}] = \frac{\hat{\pi}_g \phi(\mathbf{x}_i \mid \hat{\boldsymbol{\mu}}_g, \hat{\boldsymbol{\Sigma}}_g)}{\sum_{h=1}^{G} \hat{\pi}_h \phi(\mathbf{x}_i \mid \hat{\boldsymbol{\mu}}_h, \hat{\boldsymbol{\Sigma}}_h)} := \hat{z}_{ig},$$

  evaluated at the *a posteriori* estimates $\hat{\boldsymbol{\vartheta}}$.

- These are usually hardened to give maximum *a posteriori* (MAP) classifications.

- For example, for observation $\mathbf{x}_8$, $\hat{\mathbf{z}}_8 = (0.1, 0.3, 0.6)$ would be hardened to $\text{MAP}(\hat{\mathbf{z}}_8) = (0, 0, 1)$.

- In some applications, people prefer the soft classifications.

Hard versus Soft $\hat{z}_{ig}$

- Crucially, regardless of whether $\hat{z}_{ig}$ or $\text{MAP}\{\hat{z}_{ig}\}$ is ultimately returned, most parameter estimation approaches permit values $\hat{z}_{ig} \in [0, 1]$ as the algorithm iterates.

- This is true for the EM algorithm and the most popular alternative, variational Bayes approximations.

- The evolutionary algorithm approach that will be demonstrated later forces $\hat{z}_{ig} \in \{0, 1\}$ at all times, which is a fundamental difference between it and the more common approaches.

EM Algorithm

- The expectation-maximization (EM) algorithm (Dempster et al., 1977) is an iterative technique for finding maximum likelihood estimates when data are complete or are treated as incomplete.
- The EM algorithm is based on the idea of complete-data, i.e., observed data together with missing data.
- Recall that the Gaussian model-based clustering complete-data likelihood is

$$\mathcal{L}_c(\boldsymbol{\vartheta} \mid \mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{z}_1, \ldots, \mathbf{z}_n) = \prod_{i=1}^{n} \prod_{g=1}^{G} [\pi_g \phi(\mathbf{x}_i \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)]^{z_{ig}},$$

where $z_{ig} = 1$ if $\mathbf{x}_i$ is in component g and $z_{ig} = 0$ otherwise.

EM Algorithm contd.

- The E-step involves calculation of the expected value of the complete-data log-likelihood

$$\mathcal{Q}(\boldsymbol{\vartheta}) = \prod_{i=1}^{n} \prod_{g=1}^{G} [\pi_g \phi(\mathbf{x}_i \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g)]^{\hat{z}_{ig}},$$

where, as before,

$$\hat{z}_{ig} = \frac{\hat{\pi}_g \phi(\mathbf{x}_i \mid \hat{\boldsymbol{\mu}}_g, \hat{\boldsymbol{\Sigma}}_g)}{\sum_{h=1}^{G} \hat{\pi}_h \phi(\mathbf{x}_i \mid \hat{\boldsymbol{\mu}}_h, \hat{\boldsymbol{\Sigma}}_h)}.$$

- In the M-step, $\mathcal{Q}$ is maximized with respect to the model parameters, i.e., the parameter estimates are updated.
- The E- and M-steps are iterated until convergence.

Parameter Estimation: Convergence

- An Aitken's acceleration-based convergence criterion (Böhning et al., 1994) is used to determine convergence of the EM algorithm.
- The Aitken's acceleration at iteration k is given by

$$a^{(k)} = \frac{l^{(k+1)} - l^{(k)}}{l^{(k)} - l^{(k-1)}},$$

where $l^{(k)}$ is the log-likelihood value at iteration k.

- The asymptotic estimate of the log-likelihood at iteration $k+1$ is given by

$$l_\infty^{(k+1)} = l^{(k)} + \frac{1}{1 - a^{(k)}}(l^{(k+1)} - l^{(k)}).$$

- Use the criterion that the algorithm can be stopped when $l_\infty^{(k+1)} - l^{(k)} < \epsilon$.

The BIC

- After all members of a family are fitted, the BIC (Schwartz, 1978) is usually used to select the best model.

- The BIC can be written

$$\text{BIC} = 2l(\mathbf{x}, \hat{\boldsymbol{\vartheta}}) - \rho \log n,$$

where $\hat{\boldsymbol{\vartheta}}$ is the MLE of $\boldsymbol{\vartheta}$, ρ is the number of free parameters, and n is the number of observations.

Evolutionary Algorithm

- Consider clustering, so that all component memberships are unknown or treated as such, i.e., we do not know any of the z_{ig}.
- For the EA developed here, the fitness function is the (observed) log-likelihood, i.e.,

$$l(\boldsymbol{\vartheta}) = \sum_{i=1}^{n} \log \left\{ \sum_{g=1}^{G} \pi_g \phi(\mathbf{x}_i \mid \boldsymbol{\mu}_g, \boldsymbol{\Sigma}_g) \right\}. \tag{1}$$

- As the EA progresses, the estimated value of z_{ig} evolves.
- Use $\tilde{z}_{ig}$ to denote the estimate of z_{ig} used in our EA.

Evolutionary Algorithm

- The estimated component membership of $\mathbf{x}_i$ in our EA is given by $\tilde{\mathbf{z}}_i = (\tilde{z}_{i1}, \ldots, \tilde{z}_{iG})$ for $\tilde{z}_{ig} \in \{0, 1\}$, i.e., our labels are "hard".
- The fitness function is the log-likelihood (1) evaluated at the estimates

$$\begin{aligned} \tilde{\pi}_g &= \frac{n_g}{n}, \qquad \tilde{\boldsymbol{\mu}}_g = \frac{1}{n_g} \sum_{i=1}^{n} \tilde{z}_{ig} \mathbf{x}_i, \\ \tilde{\boldsymbol{\Sigma}}_g &= \frac{1}{n_g} \sum_{i=1}^{n} \tilde{z}_{ig} (\mathbf{x}_i - \tilde{\boldsymbol{\mu}}_g)(\mathbf{x}_i - \tilde{\boldsymbol{\mu}}_g)', \end{aligned} \tag{2}$$

where $n_g = \sum_{i=1}^{n} \tilde{z}_{ig}$.

Reproduction and Selection

- A number of single parents are used and each is cloned many times, with the cloned children reproducing as discussed here.
- For each child, i.e., each clone of a single parent, two observations are chosen at random and their $\tilde{\mathbf{z}}_i$ values are swapped.
- There is a check in the code, so that the swap only occurs if the group memberships are different, i.e., if the $\tilde{\mathbf{z}}_i$ are different; if not, other observations are selected until two different $\tilde{\mathbf{z}}_i$ are found.
- The children of the different single parents never interbreed with each other.

Reproduction and Selection

- After one instance of crossover has been carried out on each cloned child, all of the children (plus the original few single parents) are put into one list in descending order of fitness.
- The top few are selected to become the new generation of single parents.
- Similar to the CA work, this crossover procedure helps avoid stopping at local maxima of the fitness surface, i.e., the log-likelihood surface.
- However, crossover alone will not suffice in clustering applications, so a mutation step is also carried out at each iteration.
- These iterations, of crossover followed by mutation, are repeated until the EA stagnates.

Reproduction and Selection

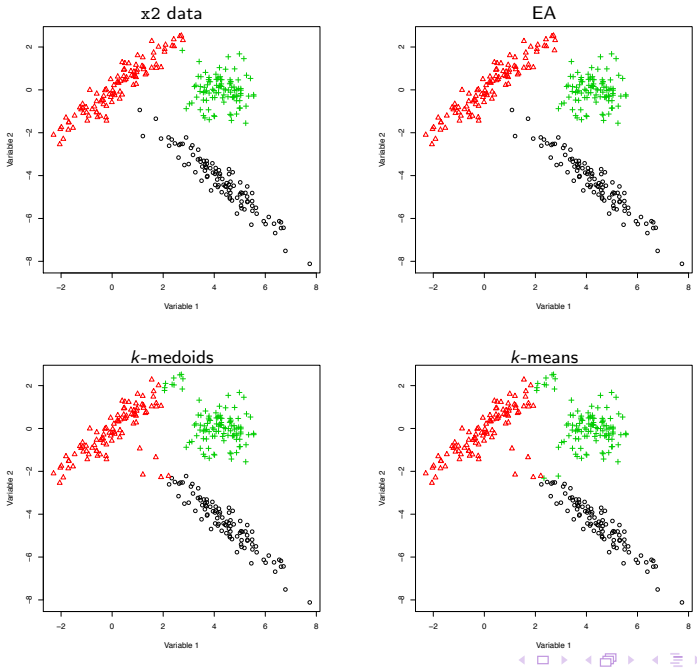
- The mutation step is "greedy"; for a given parent, once a mutation increases the fitness, the EA moves on to the next parent.
- An interpretation of the EA: the crossover step provides diversity while the mutation step allows fitness (log-likelihood) improvements that cannot be facilitated by crossover alone.
- The effectiveness of the EA for traversing the fitness (log-likelihood) surface will be illustrated next.

Pseudocode

- Very high-level pseudocode follows.

```
while stag < stagnation
  do crossover
  if list unchanged
    stag++
  else
    stag = 0
  end if else
  do mutation until fitness increases
  if no increase in fitness
    stag++
  end if
end while
return fitness values (log-likelihoods) and labels for the parents
```

Illustrations: x2 data set



Illustrations: Female Voles Data

6 morphometric measurements, as well as age, for 86 female voles from two species: *Microtus californicus* and *M. ochrogaster*.

EA	A	B
Microtus californicus	41	0
M. ochrogaster	1	44

k-**means**	A	B
M. californicus	36	5
M. ochrogaster	1	44

k-**medoids**	A	B
M. californicus	34	7
M. ochrogaster	1	44

Illustrations: Italian Wine Data

- Italian wine data: 13 chemical and physical properties of three cultivars (Barolo, Grignolino, Barbera) from the Piedmont region of Italy.
- Over all 15 runs, identical and excellent classification performance was obtained, with just one misclassification (ARI = 0.982).
- The respective classification performance of k-means (ARI = 0.897) and k-medoids (ARI = 0.741) on these data is notably inferior.

Illustrations: Italian Wine Data

EA	A	B	C
Barolo	59	0	0
Grignolino	1	70	0
Barbera	0	0	48

***k*-means**	A	B	C
Barolo	59	0	0
Grignolino	3	65	3
Barbera	0	0	48

***k*-medoids**	A	B	C
Barolo	59	0	0
Grignolino	15	55	1
Barbera	0	0	48

EA vs. EM

- Whether the EA should be directly compared to an EM is debatable because of the difference in treatment of the estimates of z_{ig}.
- This said, it is of some interest to look at relative performance.
- For the Italian wine data, the EM gives slightly inferior classification performance (ARI $= 0.945$) compared to our EA (ARI $= 0.982$).
- However, for the diabetes data, the EM gives slightly superior classification performance (ARI $= 0.664$) when compared to our EA (ARI $\in [0.584, 0.615]$).

Comments

- This EA can be viewed as an generalization of k-means clustering, in the sense that the clustering is hard but the clusters do not need to be spherical.
- k-means clustering has been shown to be equivalent to a classification EM for a Gaussian mixture model with $\boldsymbol{\Sigma}_g = \lambda \mathbf{I}_p$ (Vermunt, 2011).
- Our EA is very effective in the more general case (i.e., for a Gaussian mixture model where $\boldsymbol{\Sigma}_g$ is not constrained).