UiO : Department of Informatics
University of Oslo

Biologically inspired computing – Lecture 4

Selection
Multi-modal and multi-objective problems
Hybrid evolutionary algorithms
Working with evolutionary algorithms

This lecture

- Selection

- Multi-modal problems and diversity
- Multi-objective EAs
- Hybrid EAs
- Working with EAs
  – One-off vs. repetitive use
  – STATISTICS!

Selection

- Parent and survival selection operates independently of the representation
  - Can mix and match with representations
  - Most parent selection algorithms can also be used for survival selection by selecting μ times

Parent selection - fitness proportional selection

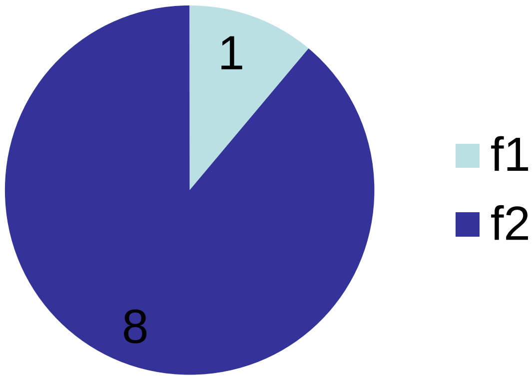
- The probability of selecting an individual is proportional to its fitness

$$p_i = \frac{f_i}{\sum_{j=1}^{\mu} f_j}$$

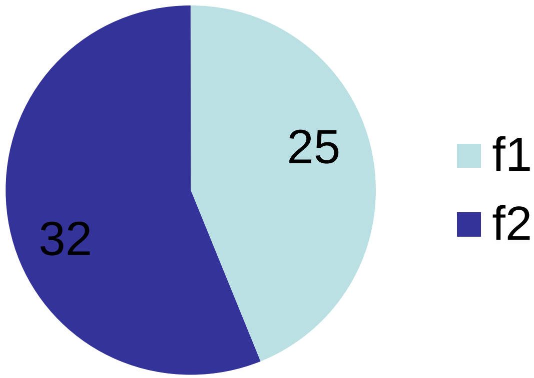
Selection pressure

- Proportional to what?
  - Selection pressure changes by adding different constant offsets to all fitnesses

Fitness near zero



Fitness far from zero



Parent selection – ranking selection

- The probability of selecting an individual is proportional to its rank

$$p_i = \frac{2-s}{\mu} + \frac{2i(s-1)}{\mu(\mu-1)}, \qquad s \in (1,2]$$

Parent selection - tournament selection

- As in evolutionary programming
  - For each parent needed, hold draw k contestants and pick the best one
  - Does not require any global information about the population
  - Gives results similar to ranking selection

Survivor selection – age-based replacement

- Few offspring compared to population size ($\lambda \leq \mu$)

- Number of generations survived → age
  - Surviving selected by age

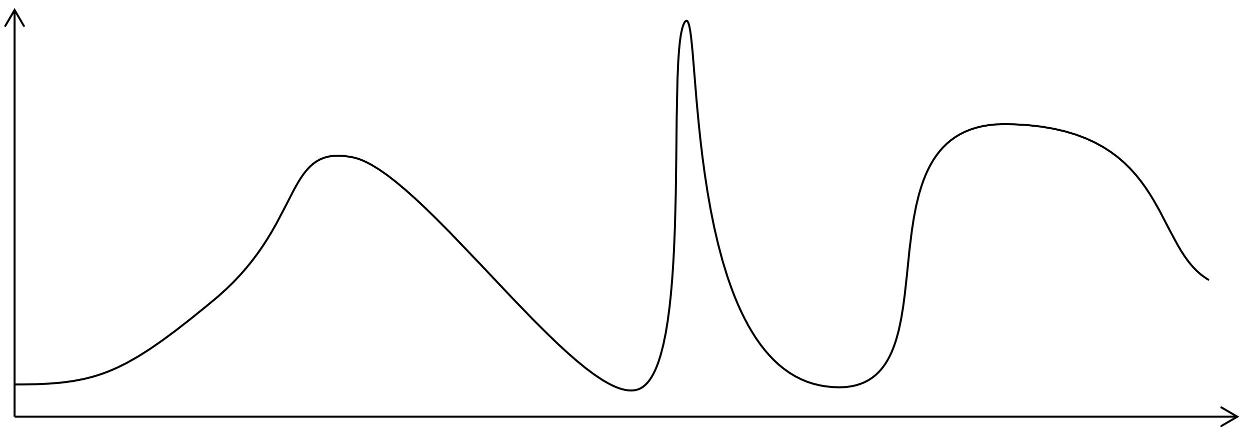
Survivor selection – fitness-based replacement

- As in evolution strategies
  - When $(\lambda \leq \mu)$: replace worst

- Elitism:
  - The very best individuals can survive indefinitely
  - Either a fixed number of elites are kept, or the number is unbounded (e.g. $(\mu + \lambda)$)

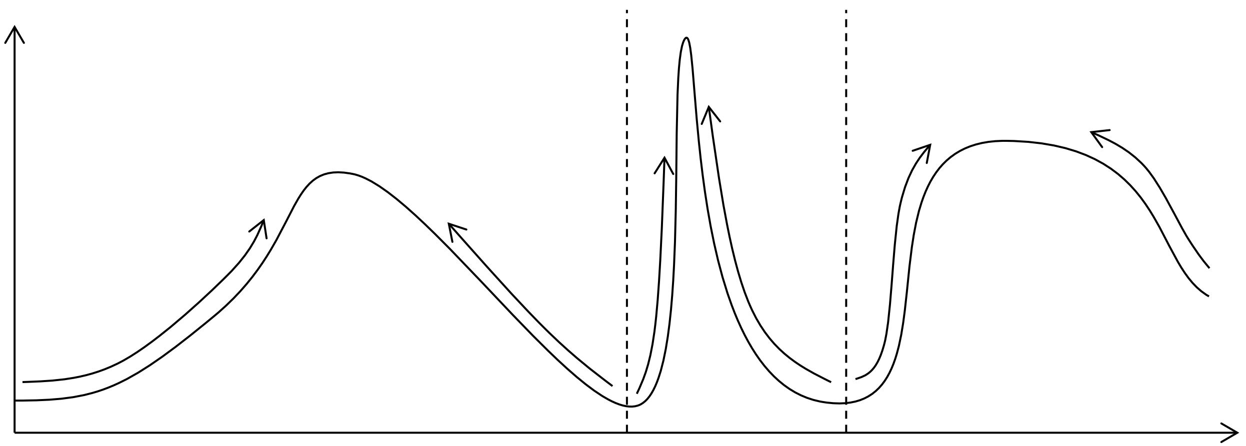
Multi-modal problems

Fitness functions usually have multiple local optima, or modes



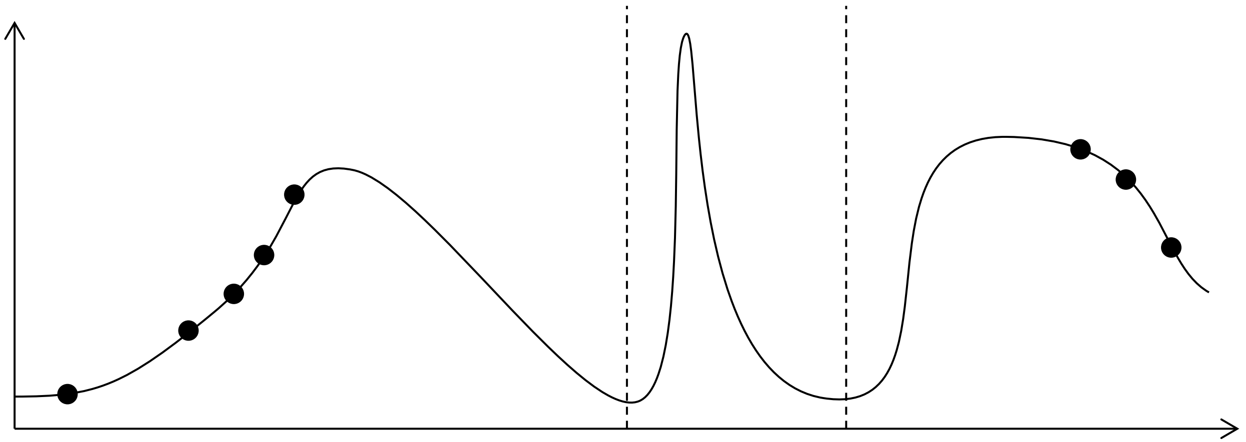
Multi-modal problems

Each of these modes will have a basin of attraction, an area around it where a local search would most likely lead to that mode



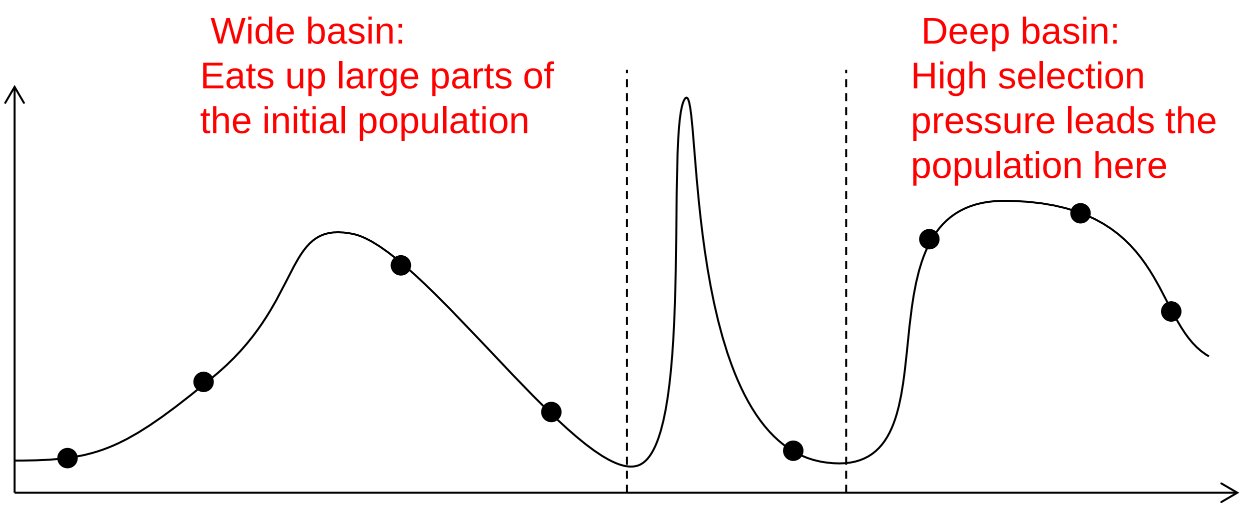
Multi-modal problems

If we're not careful with initialization, we might not get individuals in every basin of attraction



Multi-modal problems

Even when we have initialized really well there are many scenarios that can reduce the survival chances of the “right” individuals



Diversity

- Maintain individuals in as many optima as possible

- Increases the chances of getting and keeping solutions in the basin of the global optima

The island model

- Divide the population into separate “islands”
- Allow only limited migration between islands
  (e.g. a couple of individuals every 10th generation)

Diffusion model EAs

- Subpopulations have limited neighborhoods
  - Grids, rings, etc.

- Parent and survivor selection limited to the neighborhood

Fitness sharing

- Decrease the fitness of individuals with neighbors closer than σ_{share}
  - Works best with fitness proportional selection
  - Need distance measure $d_{i,j}$

$$F_i' = \frac{F_i}{\sum_j \mathrm{sh}(d_{i,j})} \leq F_i$$

$$\mathrm{sh}(d) = \begin{cases} 1 - (d/\sigma_{share})^{\alpha} & d < \sigma_{share} \\ 0 & \text{else} \end{cases}$$

Crowding

- Two parents create a pair of offspring

- Each offspring competes with their nearest parent for survival
  - Need distance measure

Speciation

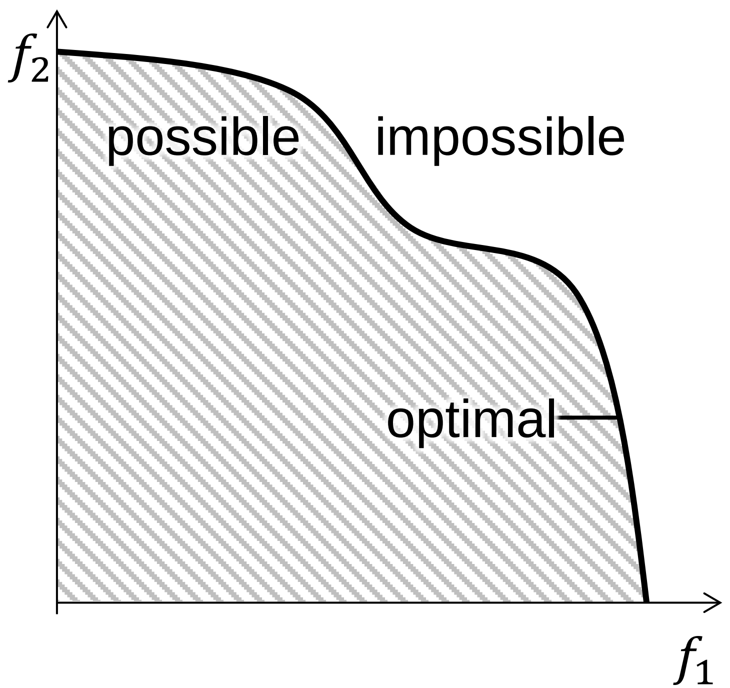
- Define subpopulations dynamically
  - Distance-based clustering
  - Genotype compatibility tags
- Restrict mating to within the subpopulations
- Subpopulation fitness sharing

Multi-objective optimization

- Conflicting considerations
  - Quality
  - Speed
  - Cost
  - etc.

Multi-objective optimization

There is no longer only one optimal solution!



Scalarization

- Use arithmetic to reduce to a single objective
- Objective priorities must be known

$$F = f_1 + af_2 + bf_3$$

$$F = f_1 + f_2 \cdot f_3$$

$$F = e^{f_1} + \tanh f_2 + \mathcal{N}(0, f_3)$$

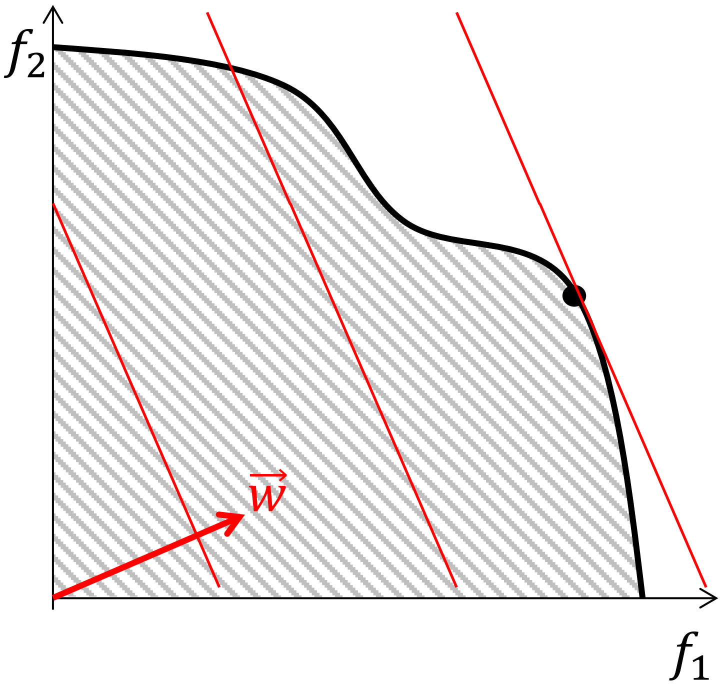
Weighted sum scalarization

- Most common scalarization
- Weight w_i is chosen based on the importance of objective i

$$F = \sum_{i=1}^{M} w_i f_i$$

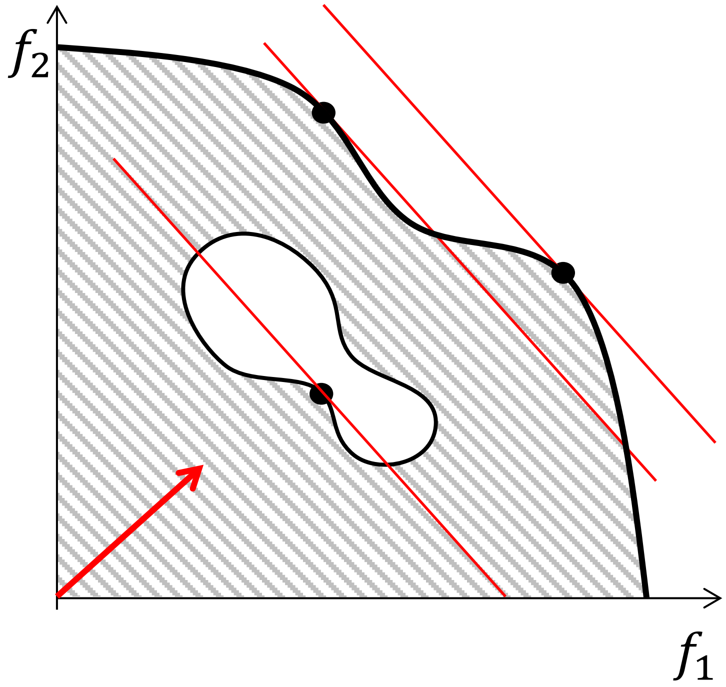
Weighted sum scalarization

The weights define a gradient in objective space



Scalarization

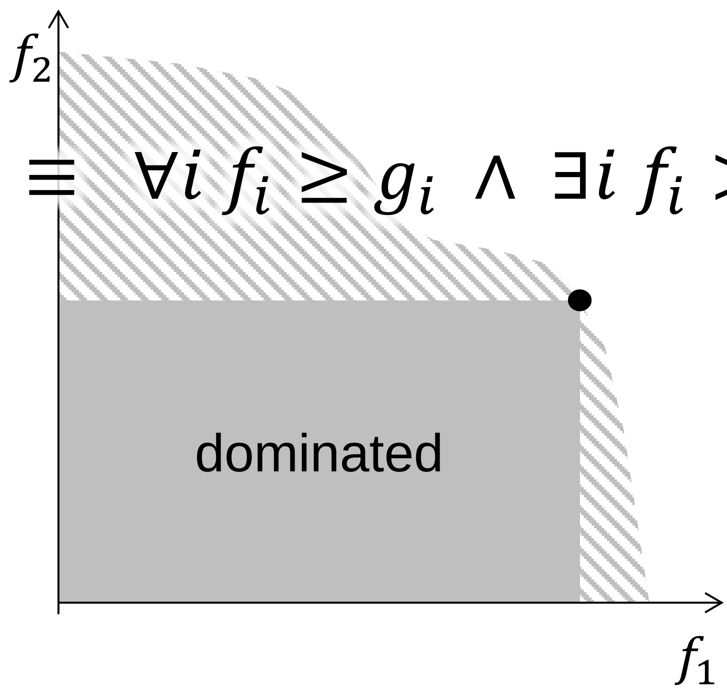
Reducing to a single objective function can create “artificial” local optima



Pareto dominance

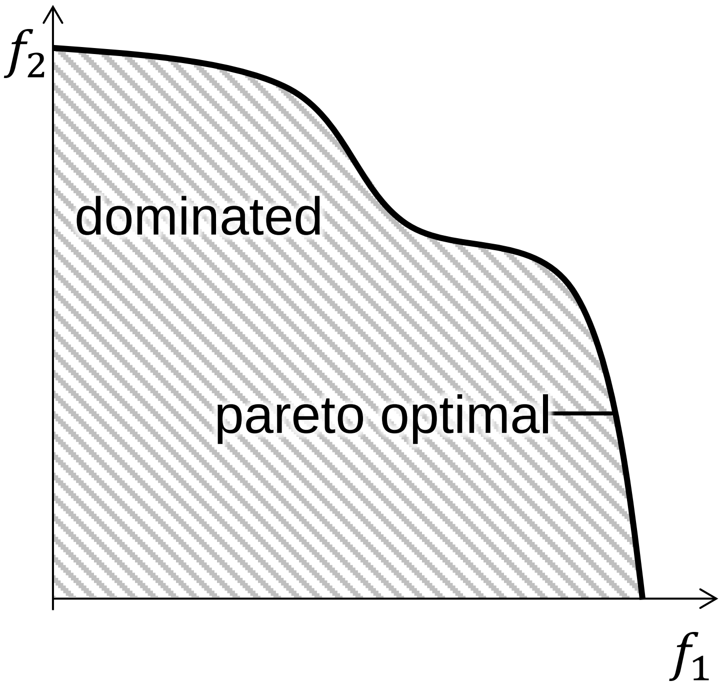
A solution dominates another if it is as good in every way and better in at least one

$$f \succcurlyeq g \quad \equiv \quad \forall i \; f_i \geq g_i \; \wedge \; \exists i \; f_i > g_i$$



Pareto dominance

Undominated solutions are Pareto optimal



Pareto dominance-based EAs

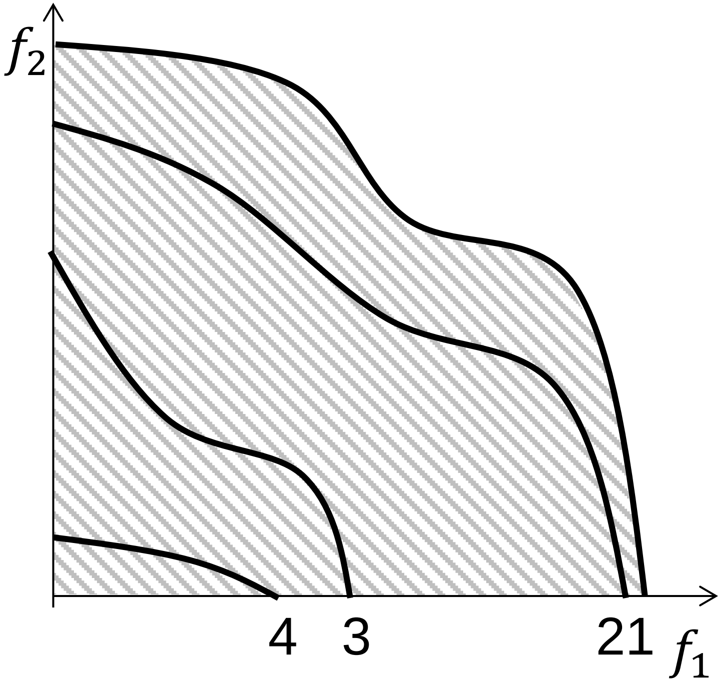
- Only selection operators are affected
- Pareto dominance replaces scalar comparison
  - Usually a secondary diversity measure is used for mutually non-dominated solutions:

```
def better_mo(a,b):
    if dominates(a,b): return true
    if dominates(b,a): return false
    return diversity(a) > diversity(b)
```

  - Tournament selection
  - Selection proportional to the number of solutions that are dominated in the population

Non-dominated sorting

```
def nondominated_sort(P):
    Q = P
    P = []
    rank = 1
    while not Q.empty():
        F = Q.nondominated()
        Q.remove(F)
        F.assign_rank(rank)
        P.insert(F)
        rank = rank+1
    return P
```



Sorts the population into layers by domination

Elitism and MOEAs

- Most modern multi-objective evolutionary algorithms use some form of elitism
  - $(\mu + \lambda)$
  - Separate population (archive)

Hybrid EAs

- In many cases, decent heuristics or local search algorithms exist

- These algorithms can often be integrated into an evolutionary algorithm either to speed up the process or improve solution quality

Hybrid EAs

Optimization	Biology
Candidate solution	Individual
Representation used in the EA	Genotype, chromosome
Problem-defined representation	Phenotype
Position/element of the genotype	Locus, gene
Local search algorithm during evaluation	Learning
Old solution	Parent
New solution	Offspring
Solution quality	Fitness
Random displacements added to offspring	Mutation
Search strategy	Mutation rate, gene robustness
A set of solutions	Population

Memetic algorithms

- Meme: An evolving cultural entity
- Another term for hybrid EAs

Memetic algorithms

- Can be added at many stages
  - Clever initialization
  - Local search during evaluation
  - Problem-specific heuristics in mutation, crossover and selection

In initialization

- Seeding
  - Known good solutions are added

- Selective initialization
  - Generate kN solutions, keep best N

- Refined start
  - Perform local search on initial population

Intelligent mutation and crossover

- Mutation bias
  - Mutation operator has bias towards certain changes
- Crossover hill-climber
  - Test all 1-point crossover results, choose best
- “Repair” mutation
  - Use heuristic to make infeasible solution feasible

Learning and evolution

- Do offspring inherit what their parents have learnt in life?
  - Yes - Lamarckian learning
    - Improved fitness and genotype
  - No - Baldwinian learning:
    - Improved fitness only

Working with evolutionary algorithms

What do you want your EA to do?

Design problems

- Only need to be done once
- End result must be excellent

Design problem example

- Optimizing spending on improvements to national road network
  - Total cost: billions of Euro
  - Computing costs negligible
  - Six months on hundreds of computers
  - Many runs possible
  - Must produce *very* good result just *once*

Repetitive problems

- Has to be run repeatedly
- Should produce OK results quickly and reliably

Repetitive problem example

- Optimizing postal delivery routes
  - Different destinations each day
  - Limited time to run algorithm each day
  - Must *always* perform *reasonably* well in limited time

Research and development

- Must produce repeatable, unbiased results
- Engineering decisions
- Academic publishing

R&D topics

- Show that an EA is applicable in some problem domain
- Show that (your fancy new) EA outperforms benchmark EA/some traditional algorithm
- Optimize or study impact of some parameters for an EA
- Investigate algorithm behavior or performance

STATISTICS

- Evolutionary algorithms are stochastic
  - Result will vary from run to run
  - Many runs are needed to say anything concrete about the performance of the EA
  - Use statistics and statistical measures!
- Do proper science!
  - Same measures
  - Fair comparisons

Things to measure

- Average result in given time
- Average time for given result
- Best result over N runs
- Proportion of X or amount of Y required to do Z under conditions W
- Etc.

Off-line performance measures

- Efficiency (speed)
  - Time
  - Average number of evaluations to solution (AES)
- Effectiveness (quality)
  - Success rate (SR)
  - Mean best fitness at termination (MBF)
    - Mean across runs, best of each run

On-line performance measures

- Population distribution (genotypic)
- Fitness distribution (phenotypic)
- Improvements per time unit or per genetic operator

What time units do we use?

- Elapsed time?
    - Depends on computer, network, etc.
- CPU time?
    - Depends on skill of programmer, implementation, etc.
- Generations?
    - Incomparable when parameters like population size change
- Evaluations?
    - Evaluation time could be small compared to “overhead”:
        - (Hybrid) selection, mutation and crossover
        - Genotype-phenotype translation

STATISTICS!

Trial	Old method	New Method
1	500	657
2	600	543
3	556	654
4	573	565
5	420	654
6	590	712
7	700	456
8	472	564
9	534	675
10	512	643
Average	545.7	612.3

STATISTICS!

Trial	Old method	New Method
1	500	657
2	600	543
3	556	654
4	573	565
5	420	654
6	590	712
7	700	456
8	472	564
9	534	675
10	512	643
Average	545.7	612.3
σ	73.60	73.55
T-test	0.0708 **NO STATISTICAL SIGNIFICANCE**	

STATISTICS!

- Don’t trust the averages
  - Extra important when the sample is small
- Always check the deviation
  - For normal distributions, use standard deviation
  - Interquartile range (1st quartile – 3rd quartile) is also good
  - Range (max – min) is better than nothing

STATISTICS!

- T-tests, use them
  - Alternatively, the simpler Z-test is approximately equivalent for good sample sizes ($N \geq 100$)
  - Different tests also exist
    - Wilcoxon
    - F-test
  - Wikipedia might be able to point out the right test