Sequential Parameter Optimization (SPO) and the Role of Tuning in Experimental Analysis

Mike Preuss
Thomas Bartz-Beielstein

Algorithm Engineering
Universität Dortmund

EMAA Workshop
PPSN 2006, Reykjavik, September 9

Overview

Is Experimentation (in EC) Scientific?

Main goal of most investigations: Comparison of optimization algorithms

How do we generate performance data?

- 2 or more algorithms, *default* parameters
- Some test problems from a standard benchmark set
- Standard performance criterion

How do we compare?

- Traditional: Compare mean values
- Since about the 90s: significance tests (e.g. t-Test)

This gets us

a) Some funny figures
b) Lots of better and better algorithms which soon disappear again

Is Experimentation (in EC) Scientific?

Main goal of most investigations: Comparison of optimization algorithms

How do we generate performance data?

- 2 or more algorithms, *default* parameters
- Some test problems from a standard benchmark set
- Standard performance criterion

How do we compare?

- Traditional: Compare mean values
- Since about the 90s: significance tests (e.g. t-Test)

This procedure appears to be

a) Arbitrary (parameter, problem, performance criterion choice?)
b) Useless, as nothing is explained and generalizability is unclear

$\Rightarrow$ Do away with experimentation?
But, in many cases, theory building also fails

Goals in Evolutionary Computation

(RG-1) *Investigation.* Specifying optimization problems, analyzing algorithms. Important parameters; what should be optimized?

(RG-2) *Comparison.* Comparing the performance of heuristics

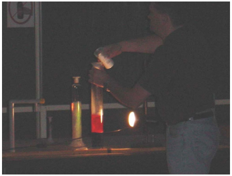
(RG-3) *Conjecture.* Good: demonstrate performance. Better: explain and understand performance

(RG-4) *Quality.* Robustness (includes insensitivity to exogenous factors, minimization of the variability) [Mon01]

Are We Alone (With This Problem)?

In natural sciences, experimentation is not in question

- Many inventions (batteries, x-rays, ...) made by experimentation, sometimes unintentional
- Experimentation leads to theory, theory has to be *useful* (can we do predictions?)
- Theory idealizes (abstraction from the real world)



This is an experiment

In computer science, the situation seems different

- 2 widespread stereotypes influence our view of computer experiments:

a) Programs do (exactly) what algorithms specify
b) Computers (programs) are deterministic, so why statistics?

Is this an experiment?

Lessons From Other Sciences

In economics, experimentation was established quite recently (compared to its age)

- Modeling human behavior as the rationality assumption (of former theories) had failed
- No accepted new model available: Experimentation came in as substitute

Nonlinear behavior

In (evolutionary) biology, experimentation and theory building both have problems

- Active experimentation only possible in special cases (*drosophila et al.*)
- Otherwise only observation (passive experimentation)
- Mainly concepts (rough working principles) instead of theories: there are always exceptions

$\Rightarrow$ Stochastical distributions, population thinking

Ernst Mayr

Current "State of the Art" in EC

Around 40 years of empirical tradition in EC, but:

- No standard scheme for reporting experiments
- Still many *horse racing* papers
- Expressiveness (task?) and reproducibility often problematic
- Experimental methodology is just forming, including new statistical tools

---

Other sciences have more structured ways to report experiments, although usually not presented in full in papers. Why?

- Natural sciences: Long tradition, setup often relatively fast, experiment itself takes time ($\Rightarrow$ results valuable)
- Computer science: Short tradition, setup (implementation) takes time, experiment itself relatively fast ($\Rightarrow$ results volatile)

Statistical Methods and Their Pitfalls

- We claim: Fundamental ideas from statistics are misunderstood!
- For example: What is the p value?

Definition (p value)

The p value is the probability that the null hypothesis is true

Statistical Methods and Their Pitfalls

- We claim: Fundamental ideas from statistics are misunderstood!
- For example: What is the p value?

Definition (p value)

The p value is the probability that the null hypothesis is true. No!

Statistical Methods and Their Pitfalls

- We claim: Fundamental ideas from statistics are misunderstood!
- For example: What is the p value?

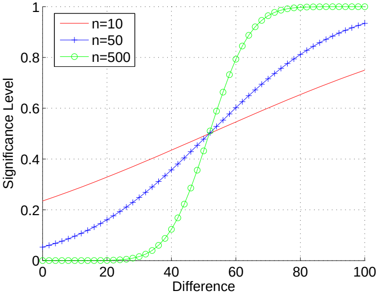
Definition (p value)

The p value is $p = P\{$ result from test statistic, or greater $\mid$ null model is true $\}$

$\Rightarrow$ The p value is not related to any probability whether the null hypothesis is true or false

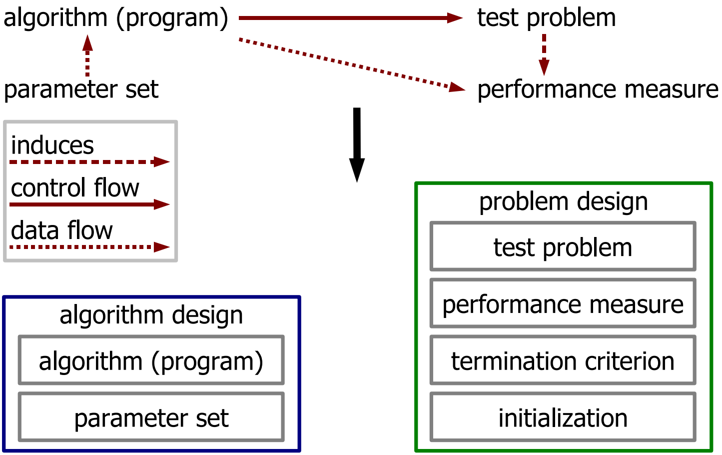
New Concepts From the New Experimentalists

- Consider scientific meaning: Largest scientifically unimportant values
- Severe testing as a basic concept
- Observed significance level (OSL) plots to support testing
- First (*higly interdisciplinary*) Symposium on Philosophy, History, and Methodology of Error, June 2006

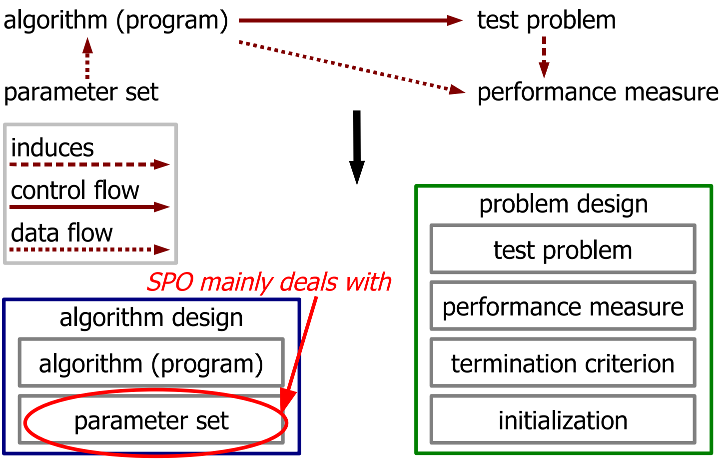


Observed Significance Level Plot

Components of an Experiment in EC



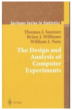
Components of an Experiment in EC



Roots and Definitions

SPO integrates elements from

Design of Experiments (DOE)



Design and Analysis of Computer Experiments (DACE) [SWN03]

- Experiment $:=$ optimization run
- Design variables / factors $:=$ parameters

- Endogenous factors: modified during the algorithm run
- Exogenous factors: kept constant during the algorithm run
  - Problem specific
  - Algorithm specific

SPO Overview

Phase I Experiment construction
Phase II SPO core: Parameter optimization
Phase III Evaluation

- Phase I and III belong to the experimental methodology (how to perform experiments)
- Phase II is the parameter handling method, shall be chosen according to the overall research task (default method is provided)
- SPO is not *per se* a meta-algorithm: We are primarily interested in the resulting algorithm designs, not in the solutions to the primordial problem

SPO Workflow

1. *Pre-experimental* planning
2. *Scientific* thesis
3. *Statistical* hypothesis
4. Experimental *design*: Problem, constraints, start-/termination criteria, performance measure, algorithm parameters

---

5. *Experiments*
6. Statistical *model* and prediction (DACE). Evaluation and visualization
7. Solution good enough?

   Yes: Goto step 8

   No: Improve the design (optimization). Goto step 5

---

8. *Acceptance/rejection* of the statistical hypothesis
9. Objective *interpretation* of the results from the previous step

SPO Core: Default Method

Heuristic for Stochastically Disturbed Function Values

- Start with latin hypercube sampling (LHS) design: Maximum spread of starting points, small number of evaluations
- Sequential enhancement, guided by DACE model
- Expected improvement: Compromise between optimization (min Y) and model exactness (min MSE)
- Budget-concept: Best search points are re-evaluated
- Fairness: Evaluate new candidates as often as the best one

Table: Current best search points recorded by SPO, initial LHS

$\frac{\lambda}{\mu}$	τ_0	*restart threshold*	*#eval best*	*config ID*	*result*	*std. deviation*
10.075	0.4180	22	4	42	0.0034	0.0058
5.675	0.7562	2	4	72	0.0042	0.0035
10.625	0.0796	5	4	57	0.0042	0.0054
4.905	0.1394	10	4	86	0.0047	0.0068
3.585	0.0398	13	4	81	0.0048	0.0056
3.145	0.0200	8	4	3	0.0050	0.0056
2.595	0.7960	4	4	83	0.0065	0.0048
2.375	1.8905	7	4	64	0.0113	0.0115

SPO Core: Default Method

Heuristic for Stochastically Disturbed Function Values

- Start with latin hypercube sampling (LHS) design: Maximum spread of starting points, small number of evaluations
- Sequential enhancement, guided by DACE model
- Expected improvement: Compromise between optimization (min Y) and model exactness (min MSE)
- Budget-concept: Best search points are re-evaluated
- Fairness: Evaluate new candidates as often as the best one

Table: Current best search points recorded by SPO, step 7

$\frac{\lambda}{\mu}$	τ_0	*restart threshold*	*#eval best*	*config ID*	*result*	*std. deviation*
5.675	0.7562	2	4	72	0.0042	0.0035
10.625	0.0796	5	4	57	0.0042	0.0054
4.905	0.1394	10	4	86	0.0047	0.0068
3.585	0.0398	13	4	81	0.0048	0.0056
3.145	0.0200	8	4	3	0.0050	0.0056
2.595	0.7960	4	4	83	0.0065	0.0048
3.866	0.0564	4	8	106	0.0096	0.0065
2.375	1.8905	7	4	64	0.0113	0.0115
...	...	...	...	...	...	...
10.075	0.4180	22	8	42	0.0177	0.0181

SPO Core: Default Method

Heuristic for Stochastically Disturbed Function Values

- Start with latin hypercube sampling (LHS) design: Maximum spread of starting points, small number of evaluations
- Sequential enhancement, guided by DACE model
- Expected improvement: Compromise between optimization (min Y) and model exactness (min MSE)
- Budget-concept: Best search points are re-evaluated
- Fairness: Evaluate new candidates as often as the best one

Table: Current best search points recorded by SPO, step 12

$\frac{\lambda}{\mu}$	τ_0	*restart threshold*	*#eval best*	*config ID*	*result*	*std. deviation*
10.625	0.0796	5	10	57	0.0024	0.0038
5.675	0.7562	2	5	72	0.0042	0.0031
4.905	0.1394	10	4	86	0.0047	0.0068
3.585	0.0398	13	4	81	0.0048	0.0056
3.145	0.0200	8	4	3	0.0050	0.0056
11.620	0.0205	2	10	111	0.0055	0.0052
2.595	0.7960	4	4	83	0.0065	0.0048
3.866	0.0564	4	8	106	0.0096	0.0065

SPO Core: Default Method

Heuristic for Stochastically Disturbed Function Values

- Start with latin hypercube sampling (LHS) design: Maximum spread of starting points, small number of evaluations
- Sequential enhancement, guided by DACE model
- Expected improvement: Compromise between optimization (min Y) and model exactness (min MSE)
- Budget-concept: Best search points are re-evaluated
- Fairness: Evaluate new candidates as often as the best one

Table: Current best search points recorded by SPO, step 17

$\frac{\lambda}{\mu}$	τ_0	*restart threshold*	*#eval best*	*config ID*	*result*	*std. deviation*
10.625	0.0796	5	20	57	0.0023	0.0034
4.881	0.0118	8	20	116	0.0028	0.0029
5.675	0.7562	2	5	72	0.0042	0.0031
4.905	0.1394	10	4	86	0.0047	0.0068
3.585	0.0398	13	4	81	0.0048	0.0056
3.145	0.0200	8	4	3	0.0050	0.0056
11.620	0.0205	2	10	111	0.0055	0.0052
7.953	0.0213	2	10	114	0.0065	0.0055

SPO Core: Default Method

Heuristic for Stochastically Disturbed Function Values

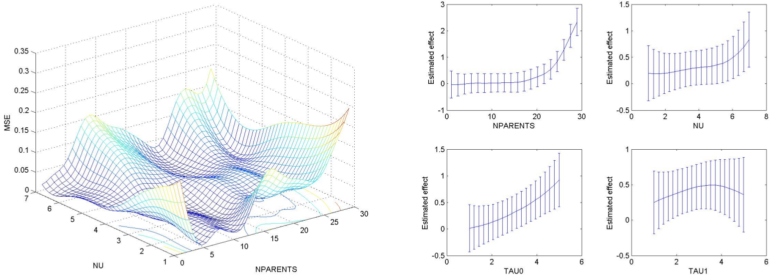
- Start with latin hypercube sampling (LHS) design: Maximum spread of starting points, small number of evaluations
- Sequential enhancement, guided by DACE model
- Expected improvement: Compromise between optimization (min Y) and model exactness (min MSE)
- Budget-concept: Best search points are re-evaluated
- Fairness: Evaluate new candidates as often as the best one

Table: Current best search points recorded by SPO, end (step 49)

$\frac{\lambda}{\mu}$	τ_0	*restart threshold*	*#eval best*	*config ID*	*result*	*std. deviation*
7.486	0.0329	13	50	140	0.0014	0.0022
6.367	0.0452	8	50	121	0.0015	0.0021
9.572	0.0536	11	50	134	0.0018	0.0031
6.024	0.0158	10	50	119	0.0019	0.0033
10.294	0.0229	8	50	133	0.0021	0.0036
6.798	0.0679	6	50	120	0.0021	0.0030
10.625	0.0796	5	50	57	0.0022	0.0032
4.8819	0.0118	8	20	116	0.0028	0.0029

SPO in Action

- Sequential Parameter Optimization Toolbox (SPOT)
- Introduced in [BB06]



- Software can be downloaded from `http://ls11-www.cs.uni-dortmund.de/people/tom/ExperimentalResearchPrograms.html`

What is the Meaning of Parameters?

Are Parameters “Bad”?

Cons:

- Multitude of parameters dismays potential users
- It is often not trivial to understand parameter-problem or parameter-parameter interactions

  $\Rightarrow$ Parameters complicate evaluating algorithm performances

But:

- Parameters are simple handles to modify (adapt) algorithms
- Many of the most successful EAs have lots of parameters
- New theoretical approaches: Parametrized algorithms / parametrized complexity, (“two-dimensional” complexity theory)

Possible Alternatives?

Parameterless EAs:

- Easy to apply, but what about performance and robustness?
- Where did the parameters go?

Usually a mix of:

- Default values, sacrificing top performance for good robustness
- Heuristic rules, applicable to *many* but not *all* situations; probably not working well for completely new applications
- (Self-)Adaptation techniques, these cannot learn too many parameter values at once, and not necessarily reduce the number of parameters

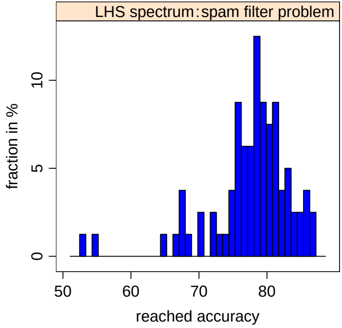
$\Rightarrow$ We can reduce the number of parameters, but usually at the cost of either performance or robustness (or both)

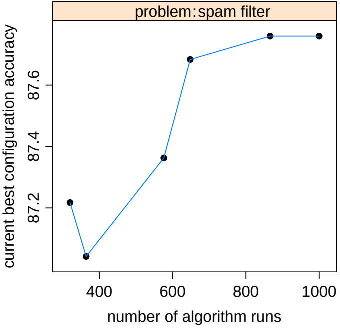
$\Rightarrow$ We probably do not get rid of several parameters in most cases

Handling Parameters: Tuning and Comparison

What do Tuning Methods (e.g. SPO) Deliver?

- A spectrum of configurations, hinting at most important parameters and parameter interactions
- A best configuration of $\{perf(alg(arg_t^{exo}))|1 \leq t \leq T\}$ for T tested ones
- A progression of current best tuning results





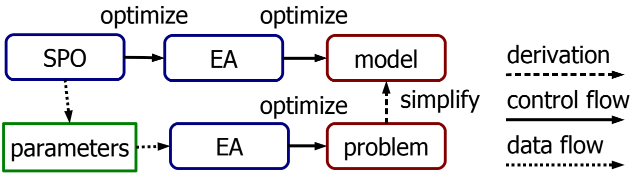
Objections Against Parameter Tuning

. . . and How to Meet them (Hopefully)

a) The meta-algorithm (1. optimize parameters of an algorithm which is 2. used to tackle the original problem) is subject to the NFL[1] (next slides)
b) Parameter optimization is too expensive

Possible solutions for b):

- Even a very small sample over the parameter space can help
- For recurring problems, parameter optimization eventually pays off
- Parameters may be optimized using simplified proxy problems (algorithm-based validation)



[1] no free lunch theorem

The Art of Comparison

Orientation

The NFL told us things we already suspected:

- We cannot hope for the one-beats-all algorithm (solving the general nonlinear programming problem)
- Efficiency of an algorithm heavily depends on the problem(s) to solve and the exogenous conditions (termination etc.)

In consequence, this means:

- The posed question is of extreme importance for the relevance of obtained results
- The focus of comparisons has to change from:

Which algorithm is better?

to

What exactly is the algorithm good for?

The Art of Comparison

Efficiency vs. Adaptability

Most existing experimental studies focus on the efficiency of optimization algorithms, but:

- Adaptability to a problem is not measured, although
- It is known as one of the key advantages of EAs

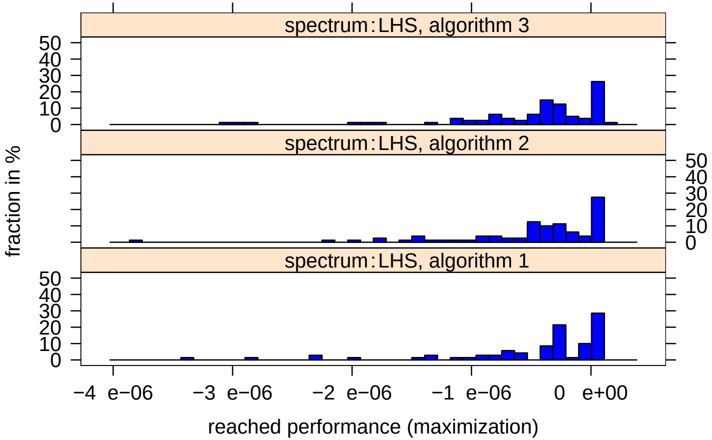
Interesting, previously neglected aspects:

- Interplay between adaptability and efficiency?
- How much effort does adaptation to a problem take for different algorithms? Or problems?
- What is the problem spectrum an algorithm performs well on?
- Systematic investigation may reveal inner logic of algorithm parts (operators, parameters, etc.)

Adaptability to a (One) Problem

Some Simple Measures

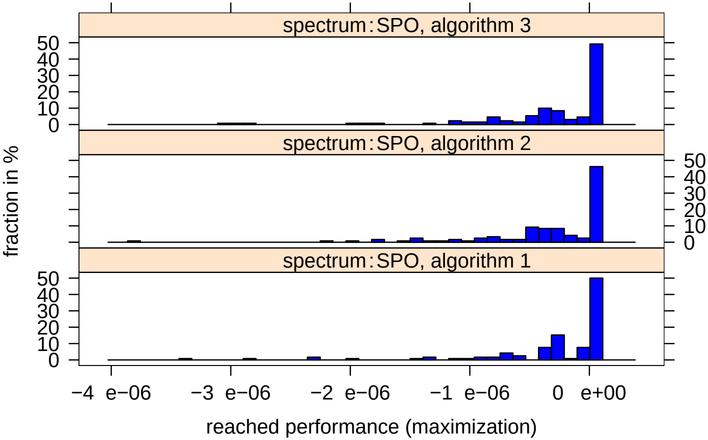
- mean(LHS(T)) $\approx$ expected performance with random parameter set
- best(LHS(T)) $\approx$ expected performance for best of random search(T)
- best(SPO(T_s)) $\approx$ performance of best existing parameter set



Adaptability to a (One) Problem

Some Simple Measures

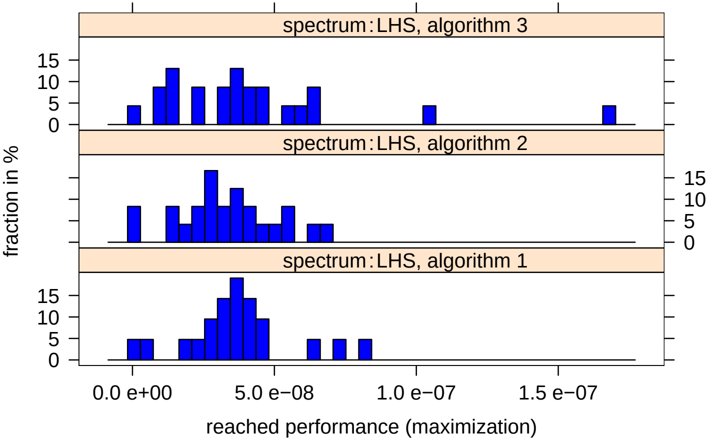
- mean(LHS(T)) $\approx$ expected performance with random parameter set
- best(LHS(T)) $\approx$ expected performance for best of random search(T)
- best(SPO(T_s)) $\approx$ performance of best existing parameter set



Adaptability to a (One) Problem

Some Simple Measures

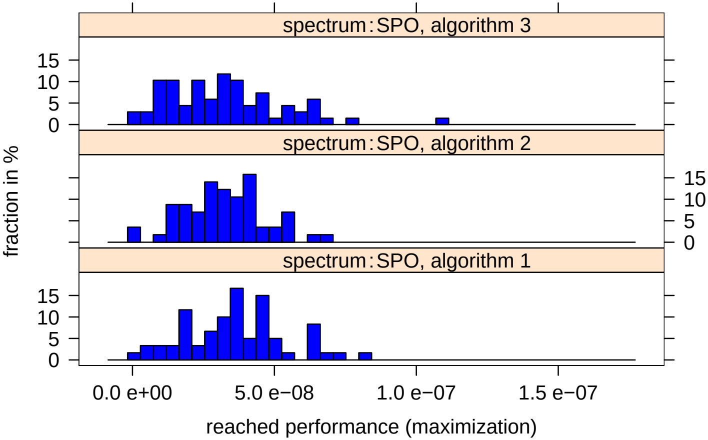
- mean(LHS(T)) $\approx$ expected performance with random parameter set
- best(LHS(T)) $\approx$ expected performance for best of random search(T)
- best(SPO(T_s)) $\approx$ performance of best existing parameter set



Adaptability to a (One) Problem

Some Simple Measures

- mean(LHS(T)) $\approx$ expected performance with random parameter set
- best(LHS(T)) $\approx$ expected performance for best of random search(T)
- best(SPO(T_s)) $\approx$ performance of best existing parameter set



Empirical Findings

Concerning the example:

- The spectra are quite similar. Are the algorithms?
- Indeed. Only the mutation adaptation operators are different.

In general:

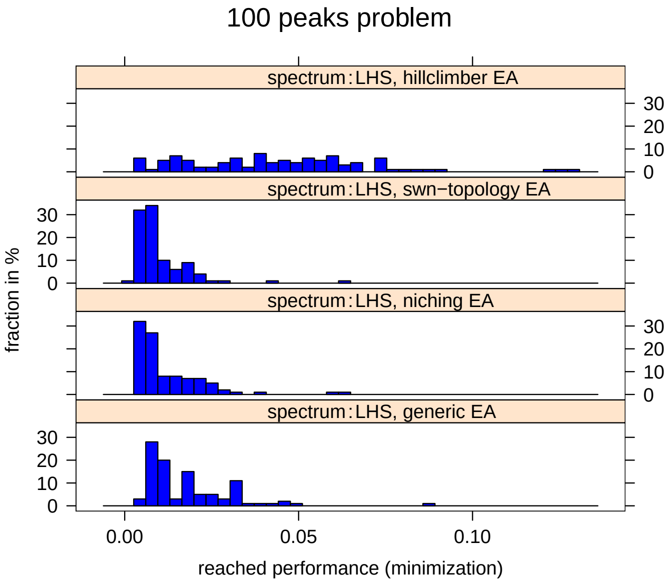
a) Some parameter sets do not work at all

b) An often found situation:
   $\frac{1}{3}$ of parameter sets lead to very bad performance
   $\frac{1}{3}$ are in the "interesting" performance region (good)
   $\frac{1}{3}$ are somewhere inbetween (not really interesting)

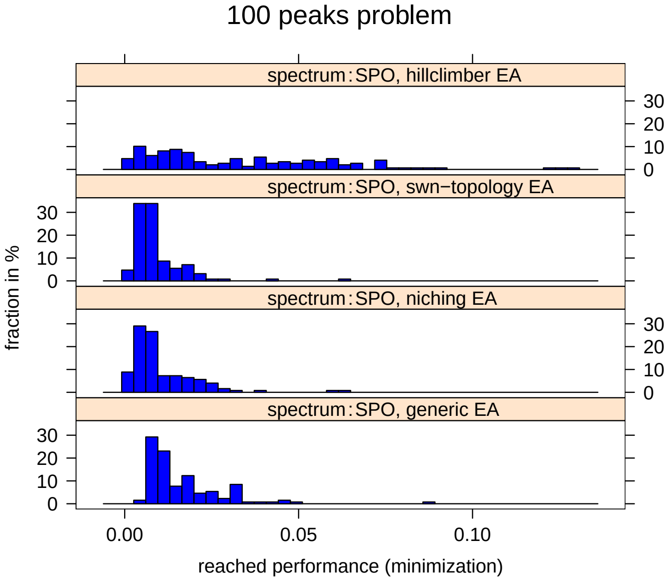
c) The performance potential SPO can reveal heavily depends on the algorithm, but with absolute distance parameters it works especially well

d) Sometimes adaptability appears to be exhausted after testing a relative small LHS design ($\Rightarrow$ low adaptability?)

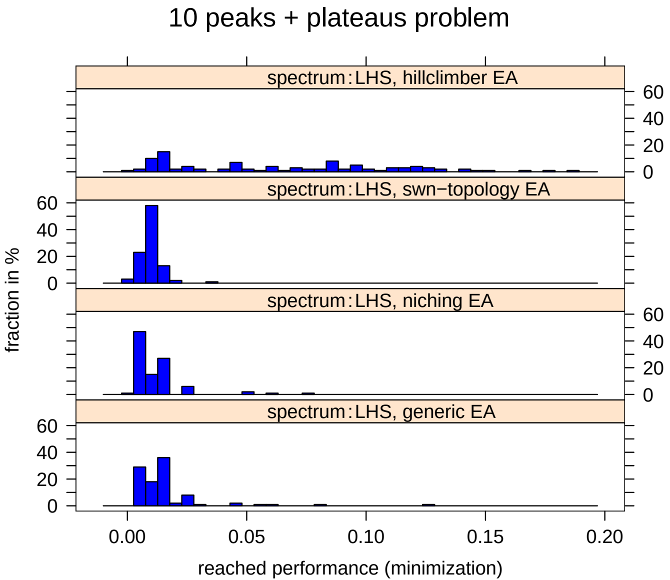
Adapting EAs to Two Related Problems



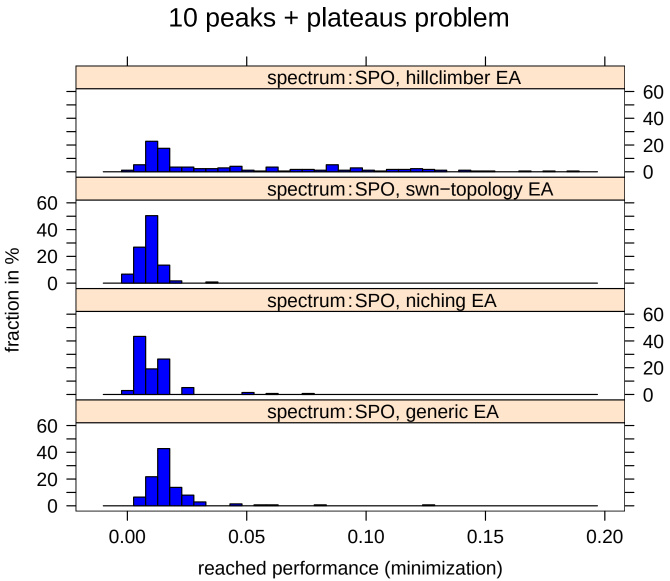
Adapting EAs to Two Related Problems



Adapting EAs to Two Related Problems



Adapting EAs to Two Related Problems



How do Tuning (SPO) Results Help?

...or Hint to new Questions

What we get:

- A near optimal configuration, permitting top performance comparison or an estimation of "adaptability potential"
- A quality estimation of any previously (manually) found parameter set

No excuse: A first impression may be attained by simply doing an LHS

Yet unsolved problems:

- How much amount to put into tuning (fixed budget, until stagnation)?
- Where shall we be on the spectrum when we compare?
- Can we compare spectra ($\Rightarrow$ adaptability)?
- How to define adaptability as a measurable size?

Thomas Bartz-Beielstein.
Experimental Research in Evolutionary Computation—The New Experimentalism.
Springer, Berlin, Heidelberg, New York, 2006.

D. C. Montgomery.
Design and Analysis of Experiments.
Wiley, New York NY, 5th edition, 2001.

T. J. Santner, B. J. Williams, and W. I. Notz.
The Design and Analysis of Computer Experiments.
Springer, Berlin, Heidelberg, New York, 2003.