



Quantile-like measures on multi-dimensional distributions of closed sets

Johann Dreoo

► To cite this version:

Johann Dreoo. Quantile-like measures on multi-dimensional distributions of closed sets. Doctoral. Innovation, Design, and Engineering, Västerås [Sweden], France. 2023. hal-04110682

HAL Id: hal-04110682

<https://hal.science/hal-04110682v1>

Submitted on 30 May 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Quantile-like measures on multi-dimensional distributions of closed sets

Application in stochastic optimization

• Johann Dreo • 2021-06-10



Abstract

- Algorithm to compute quantiles of 2D (and 3D) distributions.
- Well-founded statistics on top of them.
- Useful for multi-objective optimization problems.
- Example: automated tuning of stochastic optimization solvers.

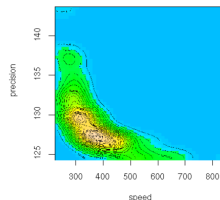


Figure 3: Projections of the performance front of the D&E method, for a setting of two mutation parameters against two objectives. Right plot shows the superimposition of the 30 performances fronts, left one shows a density estimation with gaussian kernels.

[Dre09]

Summary

01

—

Performance estimation of
optimization solvers

02

—

Quantiles on joint
distributions

03

—

Implementations

04

—

Statistics

05

—

Back to Optimization

06

—

Supplementary material

Part 1

Performance estimation of optimization solvers

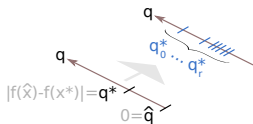
Performance(s): quality, time and probability

Optimization

q
 $|f(\hat{x}) - f(x^*)| = q^*$
 $0 = \hat{q}$

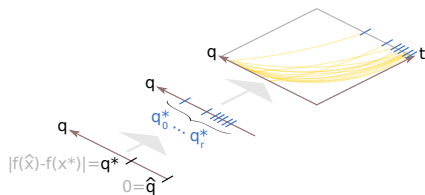
Performance(s): quality, time and probability

Terminal qualities



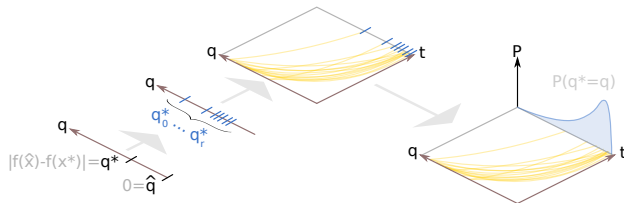
Performance(s): quality, time and probability

Convergence trajectories



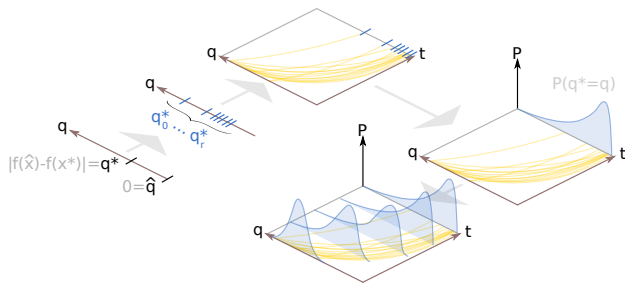
Performance(s): quality, time and probability

Terminal distributions



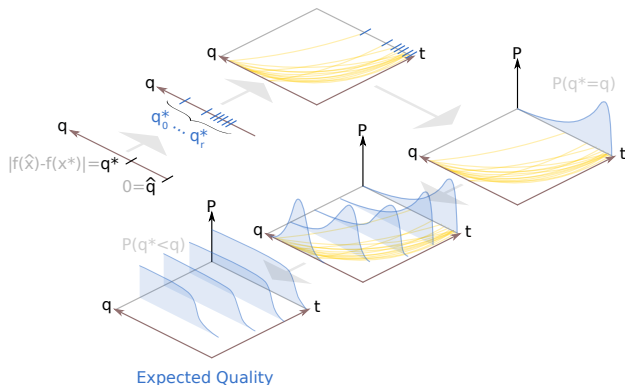
Performance(s): quality, time and probability

Convergence distributions



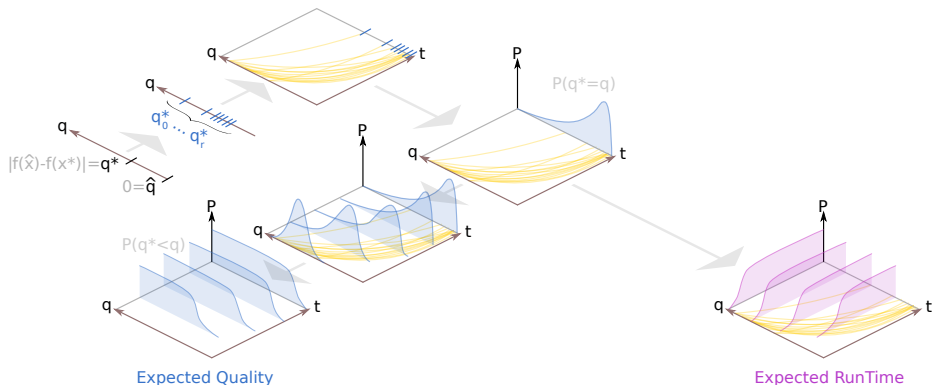
Performance(s): quality, time and probability

Expected Quality ECDF (BBComp 2015)



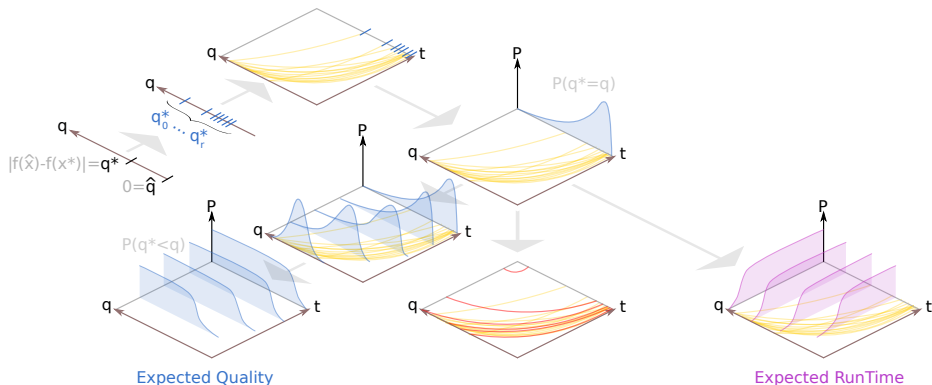
Performance(s): quality, time and probability

Expected RunTime ECDF [HAB⁺16]

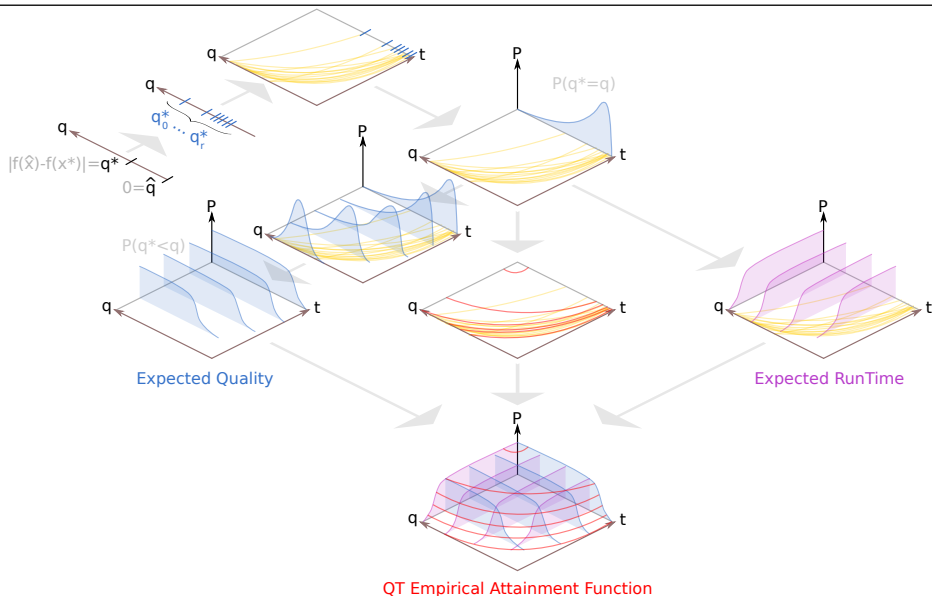


Performance(s): quality, time and probability

QT Empirical Attainment Function Levelsets [LIS14]



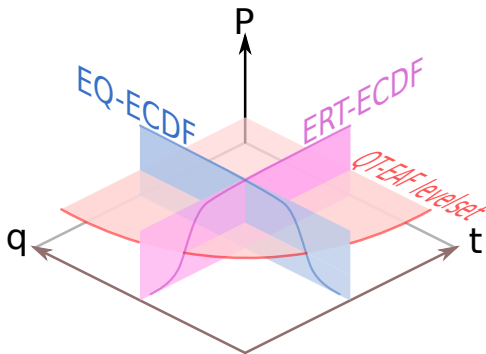
Performance(s): quality, time and probability



[Kno05]

Quality-Time Empirical Attainment Function

- EQ- and ERT-ECDF are trivially computed:
 - fix a target quality (resp. time),
 - traverse all runs across time (resp. quality),
 - compute the ratio of better-than-target.
- QT-EAF requires a more complex algorithm [GdFF02].



Part 2

Quantiles on joint distributions



Properties

Closeness

- A set of optimization trajectories forms a closed-set.
- Bounded by:
 - optimal solution, $q \in [0, bound[$,
 - time budget, $t \in [1, budget[$,
 - $P \in [0, 1]$

Properties

Monotony

- Convergence trajectories are Pareto-optimal.
- Sequence of non-dominated points^a.

^aExample from [LIS14].

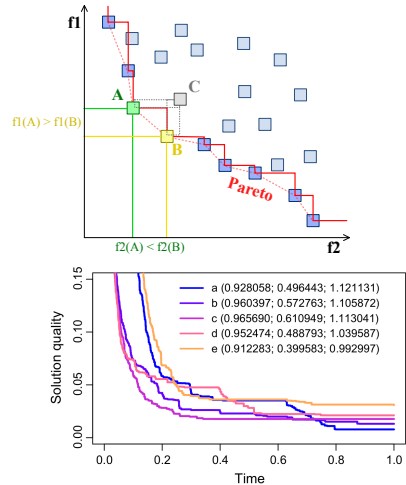
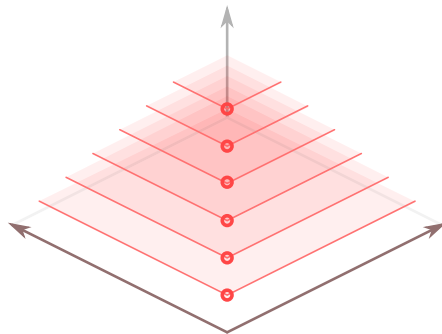


Fig. 6. For each performance profile, the legend shows the classical hypervolume, and the weighted hypervolume variants w^{qual} and w^{qual} .

Algorithm

Levelsets on finite sample

- Computes level sets of the distribution [GdFFH01].
- Which are essentially equivalent to quantiles.
- Because the sample is finite, there is a finite number of levelsets.
- At most r level sets for r input sets.



Algorithm

Example (From [LIPS10])

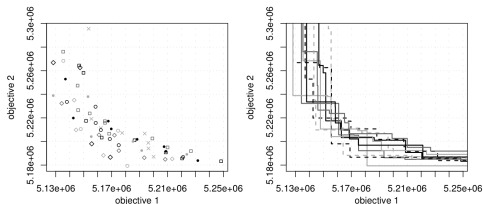


Fig. 9.1: Ten independent outcomes obtained by an SLS algorithm applied to an instance of a bi-objective optimization problem. In the right plot, the same outcomes are shown but points belonging to the same run are joined with a line

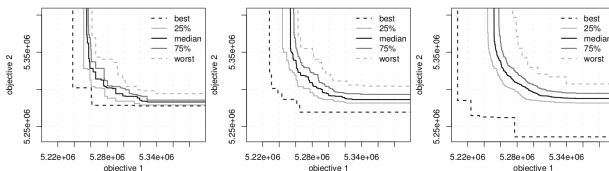


Fig. 9.3: Three plots of attainment surfaces for 15 (left), 50 (middle), and 200 (right) independent runs of the same algorithm on the same problem instance

Algorithm

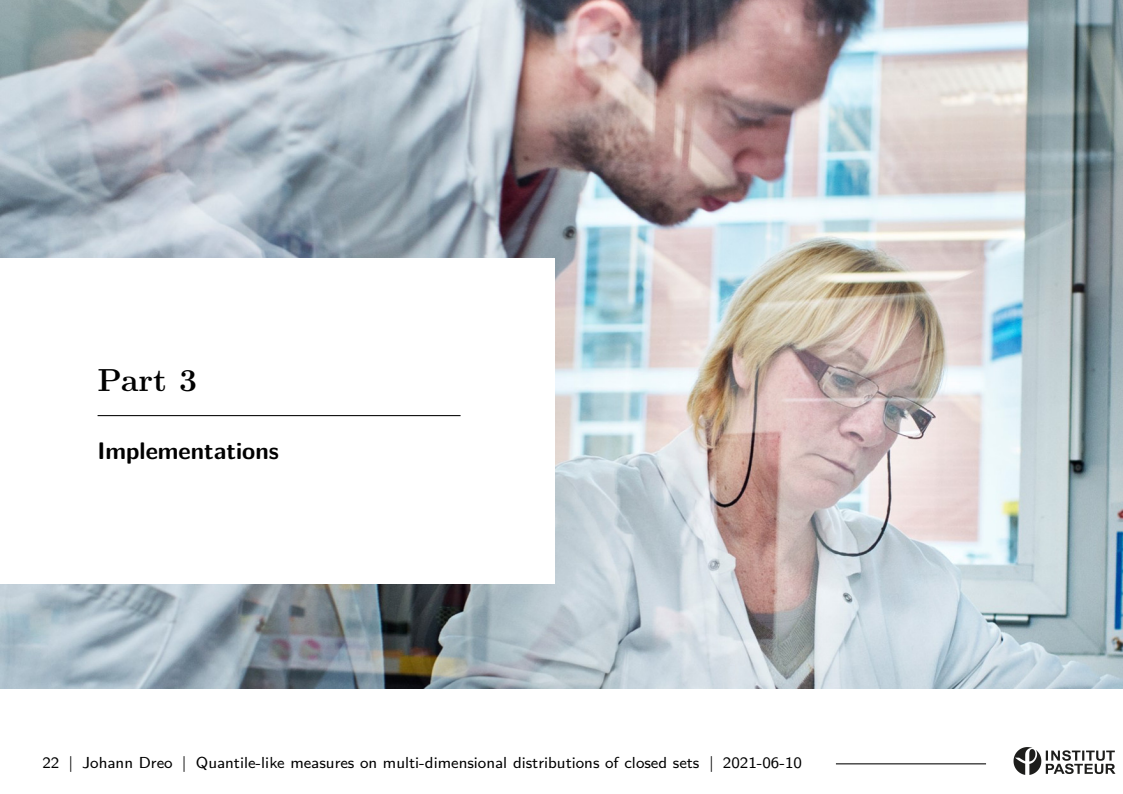
2 dimensions

- (Assuming minimization on both axis).
- “Peel” level sets.
- Sweep one axis in increasing values,
- sweep the other in decreasing values.
- Essentially computes incremental Pareto-optimal archives.
- $O(m \log m + nm)$, m points in n runs (asymptotically optimal).
-

Algorithm

3 dimensions

- $O(n^2 m \log m)$, m points in n runs,
- a logarithmic factor worse than the upper bound.
- Output *surfaces* instead of level sets.

A background image showing two scientists in a laboratory. A man in a white lab coat is leaning over a woman, also in a white lab coat and wearing glasses. They appear to be working together on a task. The setting is a modern laboratory with large windows in the background.

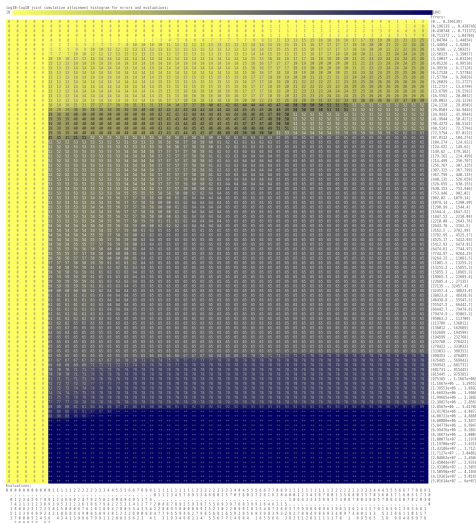
Part 3

Implementations

2D Empirical Attainment

EAF and EAH

- Two options:
 - EA *Function* (metric space)
 - EA *Histogram* (discrete buckets)



2D Empirical Attainment

Pros and cons

- If all level sets are computed, EAF is the true function (given the sample).
 - EAH is an approximation (which converges with the discretization)...
 - ... but can be computed on log-log scales, for better resolution.
-
- EAH scales better regarding the number of points...
 - ... but require more memory.

2D Empirical Attainment

- C++ implementation.
- Within IOHexperimenter
<https://iohprofiler.github.io/IOHexp/>.
- May be ported if needed.

```
492 // IOH_algorithm
493 for(const size_t level : _attainlevels) {
494     IOH_DBG(debug, "Parse level" << level);
495     std::front_max;
496     size_t it = 0; // current time index
497     size_t iq = 0; // current qual index
498     size_t nb_attained = 0;
499     // Can hold negative values.
500     std::vector<long> attained(nb_runs, 0);
501
502     // Start at upper-left corner.
503     size_t run = data_time[0].run;
504     assert(run < nb_runs);
505     attained[run]++;
506     nb_attained++;
507
508     // Result for this level.
509     std::front_level_front;
510
511     do { // while it = total and iq = total
512         IOH_DBG(debug, "Parse ascending time from" << it << " ("time" << data_time[it].time << ")",
513         // until the desired attainment level is reached.
514         while(it < total_nb_points-1 and
515             (nb_attained < level or data_time[it].time < data_time[it+1].time)) {
516             it++;
517             assert(it < total_nb_points);
518             run = data_time[it].run;
519             assert(run < nb_runs);
520             attained[run]++;
521             nb_attained++;
522             attained[run]++;
523             // if qual is equal
524             // } // while it < total_nb_points
525             IOH_DBG(debug, "Reached level at" << it);
526             // IOH_DBG(debug, nb_attained, "attained_right(n,dat)");
527
528             if(nb_attained == level) {
529                 IOH_DBG(debug, "No level is to be saved");
530                 IOH_DBG(debug, "Parse descending qual from" << iq << " ("qual" << data_qual[iq].qual << ")",
531                 // until the desired attainment level is no longer reached.
532                 do { // while nb_attained > level and iq < total
533                     iq--;
534                     assert(iq < total_nb_points);
535                     assert(iq < total_nb_points);
536                     if data_qual[iq].time < data_time[it].time {
537                         run = data_qual[iq].run;
538                         assert(run < nb_runs);
539                         attained[run]--; // can be negative.
540                         if(attained[run] == 0) {
541                             assert(nb_attained < 0);
542                             nb_attained--;
543                         }
544                         // if it is time = it, time
545                         // IOH_DBG(debug, nb_attained, "attained_down(n,dat)");
546                         // if
547                         assert(iq < total_nb_points);
548                         } while(iq < total_nb_points and data_qual[iq].qual < data_qual[iq-1].qual);
549                     } while(nb_attained == level and iq < total_nb_points);
550                     IOH_DBG(debug, "Reached level at" << iq);
551                     assert(it < total_nb_points);
552                     assert(iq < total_nb_points);
553                     IOH_DBG(debug, "No front level point!" << data_time[it].time << " " << data_qual[iq-1].qual << ")",
554                     level_front_push_back std::pair<data_time[it].time, data_qual[iq-1].qual>();
555                     // if nb_attained
556                     } while(it < total_nb_points-1 and iq < total_nb_points);
557
558                     // Save this level.
559                     IOH_DBG(note, "Level" << level << " ("nb" << level_front.size() << " points)");
560                     assert(level_front.size() > 0);
561                     assert(level_front[0].time < std::end(levels));
562                     levels[level] = level_front;
563                     // } // 1 in levels
564
565                     IOH_DBG(note, "Ended with " << levels.size() << " levels");
566                     assert(levels.size() > 0);
567                     return levels;
568                 }
569             }
```

Part 4

Statistics



Statistics for bi-objective problems

Why?

- Multi-objective problems:
 - either Pareto-optimal approaches (heavy on user),
 - either objectives aggregation (not good math properties).
 - How to aggregate randomized observations?

Statistics for bi-objective problems

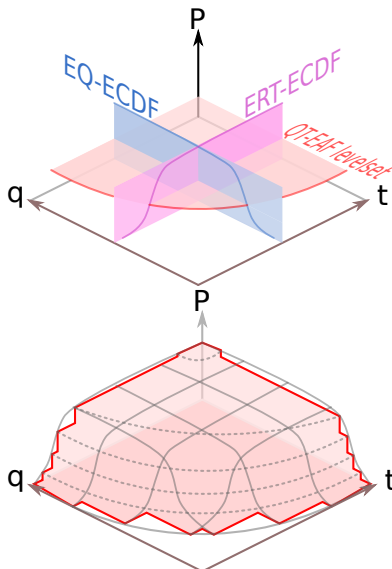
Why?

- Multi-objective problems:
 - either Pareto-optimal approaches (heavy on user),
 - either objectives aggregation (not good math properties).
 - How to aggregate randomized observations?
- EA[FH] is a way to aggregate Pareto-optimal fronts.
- One can compute statistics on it.

Statistics for bi-objective problems

Examples

- Orthogonal partial section statistics:
 - Area Under curves (EQ-ECDF and ERT-ECDF).
 - Attainment surface (EAF).
- Global statistics:
 - Volume under the EAF \approx sum/mean-like.
 - Volume under a levelset \approx quantile-like.
 - Volume under a subset of levelsets (scaling approximation).
 - Covariance [GdFFH01].



Part 5

Back to Optimization

Example of use

Automated design

- Consider the automated design of an optimization solver as a (meta) bi-objective optimization problem [Dre03].
- We want to optimize both quality and time.
- Because we have no clue about budget or target in advance.
- Maximize an average aggregate? No.
- Maximize volume under the EAF.

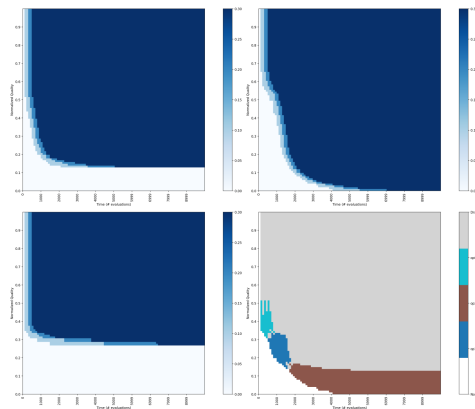


Figure 2: Dominance map for three instances CMA-ES on the BBOB Katsuura Lunacek Bi-Rastrigin function, dimension 50, first instance, with different population sizes (upper-left/blue: default population size, lower-left/cyan: 1/2 times the default, upper-right/brown: 2 times the default). White area shows the domain never attained by any algorithm, while gray area shows the domain attained by at least two algorithms with the same probability and colored area shows the domain where a single algorithm attains the higher probability.

Conclusion

Recall

- Algorithm to compute quantiles of 2D (and 3D) distributions.
- Well-founded statistics on top of them.
- Useful for multi-objective optimization problems.
- Example: automated tuning of stochastic optimization solvers.

Conclusion

Perspective

- Quantify the time/memory/loss compromises.
- Impact of the statistic choice on the optimization sub-problem.
- Use in multi-objective problems.

Time for questions

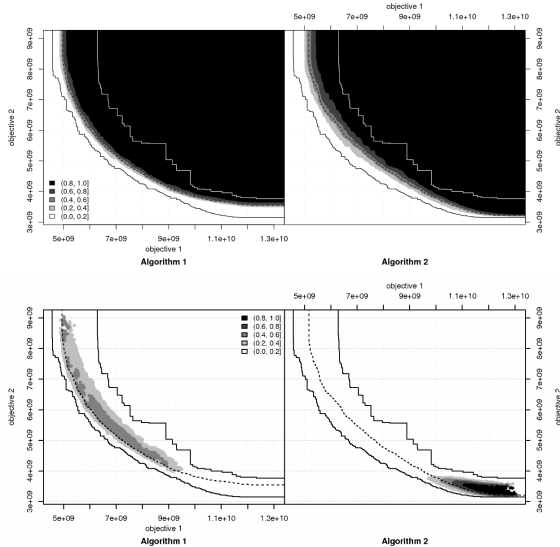
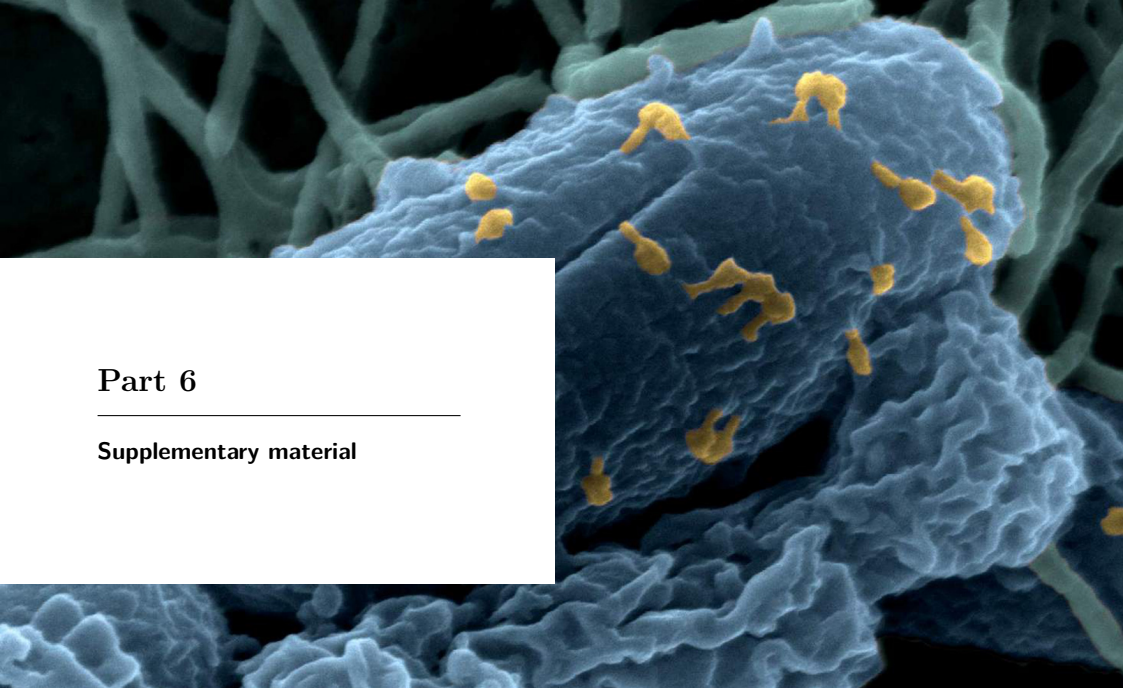


Fig. 4 Visualization of the EAFs associated to the outcomes of two algorithms (*top*) and the corresponding differences between the EAFs (*bottom left*: differences in favour of Algorithm 1; *bottom right*: differences in favour of Algorithm 2). In the top, the gray level encodes the value of the EAF. In the bottom, the gray level encodes the magnitude of the observed difference.

[LIPS10]



Part 6

Supplementary material

Bibliography



Johann Dreo.

Adaptation de la métaheuristique des colonies de fourmis pour l'optimisation difficile en variables continues. Application en génie biologique et médical.

Theses, Université Paris XII Val de Marne, December 2003.



Johann Dreo.

Using performance fronts for parameter setting of stochastic metaheuristics.

In Franz Rothlauf, editor, *Genetic and Evolutionary Computation Conference, GECCO 2009, Proceedings, Montreal, Québec, Canada, July 8-12, 2009, Companion Material*, pages 2197–2200. ACM, 2009.



Viviane Grunert da Fonseca and Carlos M. Fonseca.

A link between the multivariate cumulative distribution function and the hitting function for random closed sets.

Statistics & Probability Letters, 57(2):179–182, April 2002.

Bibliography



Viviane Grunert da Fonseca, Carlos M. Fonseca, and Andreia O. Hall.

Inferential Performance Assessment of Stochastic Optimisers and the Attainment Function.

In Eckart Zitzler, Lothar Thiele, Kalyanmoy Deb, Carlos Artemio Coello Coello, and David Corne, editors, *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 213–225, Berlin, Heidelberg, 2001. Springer.



Nikolaus Hansen, Anne Auger, Dimo Brockhoff, Dejan Tušar, and Tea Tušar.

COCO: Performance Assessment.

arXiv:1605.03560 [cs], May 2016.

arXiv: 1605.03560.

Bibliography



Joshua Knowles.

A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers.

In 5th International Conference on Intelligent Systems Design and Applications (ISDA'05), pages 552–557, September 2005.

ZSCC: 0000142 ISSN: 2164-7151.



Manuel López-Ibáñez, Luís Paquete, and Thomas Stützle.

Exploratory Analysis of Stochastic Local Search Algorithms in Biobjective Optimization.

pages 209–222. Springer, Berlin, Heidelberg, 2010.

ZSCC: NoCitationData[s0].



Manuel López-Ibáñez and Thomas Stützle.

Automatically improving the anytime behaviour of optimisation algorithms.

European Journal of Operational Research, 235(3):569–582, June 2014.

Formal concepts

Introduction

- Optimization = find the optimum \mathbf{x}^* minimizing an objective function f :

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in X} f(\mathbf{x})$$

IMG

- Randomized search heuristics approximates the optimum:

$$f(x_r^*) \leq f(\hat{\mathbf{x}}) + \epsilon$$

Formal concepts

Terminal distribution

- The output distribution of a randomized search heuristic is the probability to reach a given quality target:

IMG

$$F(q) = P[0 < Q \leq q] =$$

$$\int_0^q \lim_{r \rightarrow \infty} P[\exists r \in \mathbb{N}^+ | f(\mathbf{x}_r^*) \leq q] dq$$

Formal concepts

Temporal convergence

- Because any solver (should) be quasi-ergodic:

$$P[\mathbf{x}^* = \hat{\mathbf{x}}] > 0$$

- then it (should) converge in a finite time:

$$\lim_{t \rightarrow \infty} P[f(\mathbf{x}_t^*) = f(\hat{\mathbf{x}})] = 1$$

- Thus the probability of attaining the target should not decrease over time:

$$F(t) := P[f(\mathbf{x}_t^*) = f(\hat{\mathbf{x}})]$$

$$P[\exists r \in \mathbb{N}^+ | F_r(t-1) \leq F_r(t)] > 0$$

- Hence the trajectory in objective space of a run is monotonic.

Formal concepts

Pareto Optimality

- We say that \mathbf{u} *dominates* \mathbf{v} ($\mathbf{u} \prec \mathbf{v}$) iff

$$\mathbf{u} \neq \mathbf{v} \wedge q(\mathbf{u}) \leq q(\mathbf{v}) \wedge t(\mathbf{u}) \leq t(\mathbf{v})$$

- The trajectory in objective space being monotonic, all its points are non-dominated, and the set is Pareto-optimal.

Formal concepts

QT-EAF

- Given a set of non-dominated sets:

$$\begin{aligned} X_p = \{ & (t_1, q_1) \dots (t_m, q_m) \} \setminus \\ & \{ (t_1, q_1) \dots (t_m, q_m) \mid \\ & \nexists (t_a, q_a), (t_b, q_b) \\ & \wedge a \neq b \\ & \wedge (t_a, q_a) \preceq (t_b, q_b) \} \end{aligned}$$

- Given the indicator function $I(\cdot) : \mathbb{R}^2 \mapsto \{0, 1\}$
- The Empirical Attainment Function $EAF_r(\cdot) : \mathbb{R}^2 \mapsto [0, 1]$ is:

$$EAF_r(\mathbf{z}) = EAF_r(X_1, \dots, X_r, \mathbf{z}) := \frac{1}{r} \sum_{i=1}^r I(X_i \preceq \mathbf{z})$$

Formal concepts

QT-EAF level set

- The k level sets QAL of the EAF :

$$Z_k(\epsilon) = \{\mathbf{z} \in \mathbb{R}^2 \mid EAF_r(\mathbf{z}) \geq k \wedge EAF_r(\mathbf{z} - \epsilon) < k\}$$

$$QAL_k(\mathbf{z}) = \lim_{\epsilon \rightarrow 0} Z_k(\epsilon)$$