Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Industrial Portfolio Management for Many-Objective Optimization Algorithms

Tobias Rodemann

2018

Preprint:

This is an accepted article published in WCCI 2018 Conference. The final authenticated version is available online at: https://doi.org/10.1109/CEC.2018.8477693

Industrial Portfolio Management for Many-Objective Optimization Algorithms

Tobias Rodemann
Honda Research Institute Europe
Carl-Legien-Strasse 30
63073 Offenbach/Main
Germany
Email: tobias.rodemann@honda-ri.de

***Abstract*—In industry we see an increasing interest in (evolutionary) many objective optimization algorithms. However, the majority of engineers only using, not researching, optimizers have a limited understanding of the pros and cons of different algorithms and therefore rely on either third-party recommendations or benchmark tests to pick the most suitable methods for their problems. Unfortunately, most benchmarks are targeting an academic audience leaving the practitioner often in doubt about the correct choices. In this article we try to outline the essential requirements for a many-objective optimization algorithm portfolio management from an industrial perspective and compare the situation in our field to another domain with similar issues, image processing. We want to address one of the core practical issues: "Given a limited computational or time budget for my optimization project, which optimization algorithms should I try?".**

I. Introduction

An increasing complexity of technical systems has led to a growing interest in algorithms for many-objective optimization (MAO), that is problems with more than three objectives. The academic community has already responded to this challenge and published a large number (50+,[1]) of MAO algorithms.

At our lab we have been researching and applying evolutionary algorithms for about two decades, which gives us a good insight into both academic and applied usage of EAs (Evolutionary Algorithms). We have done numerous optimization projects for, among others, turbine engines [2], micro heat exchanger [3], crash safety [4], or hybrid cars [5].

For single objective optimization problems we have early on decided to use the CMA-ES [6] as our main optimizer, due to its robustness and good performance on most of our problems. This optimizer is also used intensively by other research groups, see for example [7].

Unfortunately, for many-objective optimization the situation is more difficult. For example, we were recently asked by a colleague from one of our development units: "We have a many-objective optimization problem - which algorithms should we use?". A simple question, but difficult to answer. The optimizer performance could have a serious impact affecting possibly millions of customers.

Providing a good answer would mean that we suggest a list of algorithms that maximize the chance of finding good solutions considering a limited computational or time budget. This decision would have to be based on limited information about the application problem (like number of parameters and objectives), maybe some initial tests (for example Design-of-Experiments, DoE), and results from several, mostly academic benchmark tests.

The obvious solution would be to consult a benchmarking study and use the best ranked algorithm. However, as we will see below, benchmarks in our field currently do no provide clear guidelines for portfolio management. It is not even clear whether newer algorithms really provide some benefits over older ones. The majority of benchmark studies does not provide a clear conclusion (i.e. a ranking of algorithms) so that it is up to the end-user to decide which algorithm to employ.

Benchmarking is an essential part of algorithm development used to assess the merits and de-merits of different methods on a variety of test cases. At the same time it is probably one of the least popular (and least creative) activities in optimization research. As a result efforts for benchmarking are often minimized to satisfy reviewers demands and highly constrained by the available time (for example until the next paper deadline). The problem of benchmarking for MAO problems is also being discussed in [8].

Unfortunately, the majority of benchmarks from the literature were designed with a clear research target, but are difficult to use for an engineer. In the next sections we are going to outline why these tests are suboptimal for industry purposes. Before that, we are going to give a short introduction to many-objective optimization.

We are targeting an efficient portfolio management of MAO algorithms, trying to satisfy various performance and flexibility requirements with a minimum maintenance effort. This is because finding, implementing, documenting, assessing, and maintaining a large number of algorithms is very costly. We are therefore very much interested in good benchmarks that deliver the necessary information for deciding which algorithms to maintain and use.

At this point we want to concede that this paper will not present any results, solutions, or answers. Its main point is to raise the awareness that something fundamentally important is missing in our community.

Accepted for IEEE WCCI-CEC, July 8 – 13, 2018 in Rio de Janeiro, Brazil
©2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

II. Many-objective algorithms, benchmarks and performance indicators

To give the reader from a more general evolutionary algorithm background a better understanding about the specific class of problems we currently intensively work on, we now shortly outline the field of many-objective optimization.

There are mainly two different sub-fields dealing with the challenges of a larger number of objectives. The first field is Many-Objective Optimization (MAO), where many is generally [9] understood as meaning 4 or more. The other field is Multi-Criteria Decision Making (MCDM) where the selection of the most suitable solution (in the eyes of a human Decision Maker (DM)) is in the focus. A link between the two fields is the approach of preference-based optimization, where the DM has already expressed a preference for solutions, which would enhance both the MAO and the MCDM process.

A. Many-objective Optimization

With an increasing number of objectives, finding the optimal set of solutions, an approximation of the Pareto set, becomes more and more challenging. For most problems selection pressure decreases, i.e. the majority of solutions is not dominated by or dominating any other solution. This issue can be tackled by indicator-based approaches like SMS-EMOA [10] or HypE [11], modified selection operators as in CDAS [12] or the decomposition of the many-objective problem into several smaller and easier to solve problems. Examples for the latter are MOEA/D [13], RVEA [14], or NSGA-III [15]. Decomposition-based approaches often deliver a good performance on many tests, but substantially increase the complexity of the algorithm making them very difficult to use by non-experts. At the other end, the SMS-EMOA is conceptually very simple, often has a good performance but might slow the optimization process down due to hypervolume computation. For more detailed comparisons of MAO algorithms consult [16], [17].

From a practical perspective the progress of a many-objective optimization is very difficult to assess and the found solutions are either too sparsely distributed in objective space to allow a good understanding of the Pareto front or are too numerous to test all of them in detail.

In many applications, selecting the best suited from the final solutions of an optimization run might be more difficult than finding those solutions. If no explicit weighting factors are available (a problem especially in applications with objectives from different domains, like [5]) all solutions might be valid candidates.

In MCDM a computer program tries to support this decision making process by improved visualization, functions to narrow down the set of solutions or estimation of objectives weighting factors. See [18] for some examples of MCDM visualization tools.

B. Preference-based optimization

A promising solution to the problems of MAO and MCDM is preference-based optimization. The basic idea is to provide the MAO algorithm with an ideal solution (a definition of objective values or a weighting of objectives). These preferences could then be used to focus computational resources onto a much smaller area of objective space, providing a natural direction for selection (towards the preference) and simplifying the MCDM problem. Ideally, the DM would just take the solution which is closest to the preference solution.

Preference-based MAO is a rather new field, but in recent years a number of algorithms have been proposed that can incorporate preferences with no or only minimal modifications (e.g. [13], [15], [14]). These algorithms are decomposition-based and simply adapt the problem decomposition to the expressed preferences. An alternative approach is to use desirabilities [19], [20] that can also be used to express preferences via desirability functions and easily interface with any MAO algorithm, for example the SMS-EMOA [10].

Preference-based optimization is very promising for industrial applications since in many real-world problems, there are reference systems, for example a previous version of the product. In [20] we tested the usage of desirabilities and in [16] we compared three decomposition-based preference algorithms on this problem. The results were very encouraging but also demonstrated that doing a benchmark on preference-based optimization methods is not easier but actually more complicated than standard (global) optimization. This is because it is not even clear what type of result relative to the expressed preferences would be preferred, as many solutions as close to the preferences as possible or a large variety of solutions in a larger vicinity.

We conclude that from an application point of view preference-based optimization is very promising but there are many unsolved issues, making these methods still a bit difficult to apply in practice.

III. Issues with Current Benchmarking Approaches

In academia there is typically a small number of standard benchmark suites used for assessing the performance of an algorithm. In the field of MAO the main benchmarks are DTLZ [21], WFG [22], ZDT [23]. These test functions have a number of beneficial characteristics which make them so popular:

- The test function is explicitly given as (simple) equations.
- In most cases optimal solutions, Pareto fronts, optimal HV and IGD values can be computed analytically.
- Most test functions are scalable in the number of parameters and objectives, so the performance of the algorithm can be evaluated for different problem sizes.
- Computation time is very low allowing for many generations, large populations and excellent test statistics.
- It is possible to analyze and understand the type of the optimization problem (and adapt the test function to show a specific behavior).

For many real-world problems most of the above features are not true. Fitness computation is slow and (at least initially) no information on the nature of the problem is available. It is

also often not trivial to find out which of the many academic benchmarks is similar to the specific application problem and therefore of practical relevance.

Another problem with academic benchmarks is that they tend to be too simple. Typically parameter or objective ranges are limited (e.g. from [0,1]) and all parameters and objectives tend to be structurally similar. With that we mean that they all have similar ranges, a similar type of non-linearity and all parameters have a similar impact on objective values. In real-world problems, parameter ranges might differ substantially (and normalization can be tricky), some parameters might only have a minor impact on a few objectives while others might have a dominant impact on most objectives. It is also possible that some parameters only matter if other parameters are in a certain range. Problems of this type might be difficult to assess with academic benchmarks.

Almost all academic benchmark problems have an identity mapping between genotype and phenotype space. This is rather untypical in real-world applications, where genotype parameters are used to build the system phenotype in a complex, highly non-linear process, as for example in shape optimization where design parameters represent control points on a spline or local deformations of an initial shape [2]. In another class of problems design parameters specify controllers [24], [5] that determine the performance of a system. Objectives are then computed based on the performance of the controlled system in a specific environment.

Most benchmark functions have a negligible computational demand compared to real-world applications, where a single fitness evaluation might take hours. Therefore the relative weighting of higher algorithmic complexity is very different. A typical example is the SMS-EMOA for which the cost of hypervolume computation is often considered to be excessively high. However, this would strongly depend on the relation between number of objectives and population size on the one hand and fitness evaluation time on the other hand. Algorithms might be sorted out due to seemingly too long run-times when this is actually no problem in practice where fitness computation takes very long.

Finally, there is a practical problem when testing a specific implementation of an algorithm using academic benchmark suites. While the algorithmic concept of an optimizer might have no problem with very small or very large parameter or objectives values, a mix of positive and negative values, or large differences in values, the actual implementation (software) might not be that robust. It can therefore happen that an algorithm (software) that runs perfectly on academic benchmark functions might fail on real-world problems, for example due to some numerical instability for large parameter values. Having a wider variety of test cases could therefore help to improve software testing to deliver well-tested algorithms that might actually work off-the-shelf.

A. Ranking

Ideally, there is a clear guideline how to choose the best algorithm, depending on either

1) a direct ranking of algorithms based on a larger number of relevant benchmarks
2) clear instructions on how to map known features of an application optimization problem (for example number of parameters or number of objectives) to a decision about the best algorithm

It has to be considered that industrial optimization tasks are often computationally very demanding, therefore only a very limited number of optimizers can be tested. Trying a dozen algorithms including a decent hyperparameter scan [25] is rarely an option.

For various reasons, most authors of benchmark papers hesitate to clearly state which algorithm is best based on their findings. Approaches like TOPSIS [26] might be used to provide an explicit ranking of optimizers over several benchmark functions, which would probably lead to some intense arguments over the published results. However, this type of ranking would make the selection of algorithms much easier for engineers.

B. Selection of a Portfolio

It is probably unrealistic to expect a single approach to be optimal for all applications. More likely, different approaches would be optimal on different types of problems. The target of the portfolio management is therefore not to select the, say, three best optimizers according to some performance ranking, but to select three algorithms which together maximize the probability of finding good solutions. Therefore one would be interested in finding algorithms that are rather orthogonal in their performance (one is strong when the others are weak).

C. Hyperparameter Tuning

Most benchmark tests use a single set of hyperparameters, i.e. the parameters of the optimizer itself, like population size, mutation rates, crossover probabilities and so on. This is often the hyperparameter setting used by the original authors (due to the lack of any better choice). It is often unclear to which degree performance might change (for better or worse) if these hyperparameters are chosen differently. Recently automatic meta-optimization approaches like irace [25] have been proposed, but are not yet used widely. In any case, the only realistic usage of these approaches is on the side of benchmarks tests by algorithm developers, since on the engineering side theses tests would take too much time. For practical applications this is a very relevant factor. For a naive user, a proper setting of several critical hyperparameters is virtually impossible. It is therefore essential that the optimizer still works under sub-optimal settings.

A very promising step is the automatic adaptation of the optimization algorithm itself, as in the Borg approach [27]. It is to be seen if Borg can consistently compete with other algorithms. Since Borg first needs to adapt itself to the problem, requiring a decent number of function evaluations, it might not be efficient when only very few fitness evaluations are possible.

D. Alternative facts

Benchmarks or competitions in general are always in danger of getting rigged. Currently authors are free to compare the algorithms and benchmark functions they like (with occasional additions from the reviewers). There is obviously a strong temptation for motivated researchers to optimize their benchmark tests by selecting (or reporting) the most suitable test functions and competing algorithms to make their own method look as good as possible.

Considering that algorithmic performance varies with benchmark problem it is quite likely that the ranking of different algorithms can be inverted by clever selection of benchmark function and competing algorithms. Assuming that we want to produce evidence that our new algorithm outperforms some previous ones, we might compare our methods to say 10 different competing algorithms on 10 different benchmarks. Then we could pick the combination of a few benchmarks and a few algorithms with the worst performance and would probably have a decent chance of proving the superiority of our algorithm.

Kenneth Sörensen, in his fascinating keynote talk at EvoStar 2016, noted that several benchmark tests his group investigated were so blatantly fake that it was actually possible to prove that the results are impossible to reach. He noted that this phenomenon is closely aligned with another big problem, so called metaphor-based methods (see [28]), that are flooding the relevant literature with low quality and low innovation algorithms, motivated only by a metaphorical resemblance to some phenomenon from nature, technology or science.

As an example, we recently came across a review paper on optimization methods for building performance optimization [29]. The authors provide a broad overview on various challenges in optimization including multi-objective optimization. But when looking at the list of mentioned algorithms we find everything from the standard NSGA-II and Particle Swarm Optimization (PSO) to "Intelligent Water Drop", "Harmony Search", "Firefly algorithm", and "Invasive Weed Optimization Algorithm". The latter are unlikely to be in the top 10 algorithms.

Even if attempts are made to conduct a fair test, without a clear ranking (or a list of generally agreed on high-performance algorithms) authors will conduct their tests using competing algorithms they consider as standard, which is highly subjective.

In addition to manipulation of benchmark tests, there is also the issue of software bugs which might lead to wrong conclusions. Even in standard, publicly available software packages there might be severe bugs (as we found out in our own tests), that would totally invalidate any benchmark result.

E. Commercial optimization tools

One would assume that the developers of commercial optimization tools like modeFrontier [30], OptiY [31], Matlab Global Optimization toolbox [32], pSeven [33], and many others would be among the main users of benchmark test results. Ideally, the optimization methods that ranked best in recent tests should be included in these tools to provide optimum performance. At least a quick look at the available methods does not confirm this. Many popular methods in academia are not implemented in those tools, and methods from commercial tools are not benchmarked in standard tests. It is unclear whether this is due to licensing issues, missing demand from customers, or whether this is related to the low relevance of academic benchmarks for industrial applications (i.e. the tool vendors' customers). In any case, the limited attendance of software companies on conferences in our field is an indicator that the transfer of knowledge between fundamental research and application does not work in an optimal way. We believe that the development of more relevant benchmark suites should also try to integrate the major suppliers of optimization software tools for industry. They would be able to define a broader spectrum of relevant test applications based on customer requests. It is also interesting to note that several commercial tools provide automatic support for selecting the best suited optimization approach (from a large range of methods even beyond the evolutionary computation field). It would be highly inspirational to understand how our methods are ranked in this comparison and where they are perceived to fail.

F. Digital Database

We consider current benchmarking practices to be somehow inefficient for the scientific community as a whole and also for single users. Too many research groups develop their own software for optimizers, performance indicators, benchmarks, and statistical test analysis. Many tests of popular optimizers on common benchmarks (take the performance of a common approach like NSGA-III on a standard benchmark as an example) have probably been repeated (and published) dozens of times. A better re-usability of previously published results (e.g. a common database) would massively reduce the effort for benchmark tests, reduce paper length and allow a stronger focus on relevant new findings. Obviously this would have to be balanced with the need to have some redundancy in the tests.

We would also argue that publishing benchmarks on paper is a poor way of organizing this. It is tedious and error prone to copy data from a paper into a local database or software (and vice versa). Furthermore benchmarks are, given the rapid pace in research, quickly outdated. Papers are not easily searchable and the raw data is often not available. We believe that benchmarks should be organized using well-maintained data bases, that are kept up-to-date and provide the necessary data in various formats for further processing. In pure digital form substantially more information can be provided than will ever be possible within a paper.

G. The State of Benchmarks

Looking at all the issues mentioned above it has to be concluded that the majority of academic benchmark studies is not designed to allow somebody who is not an expert in

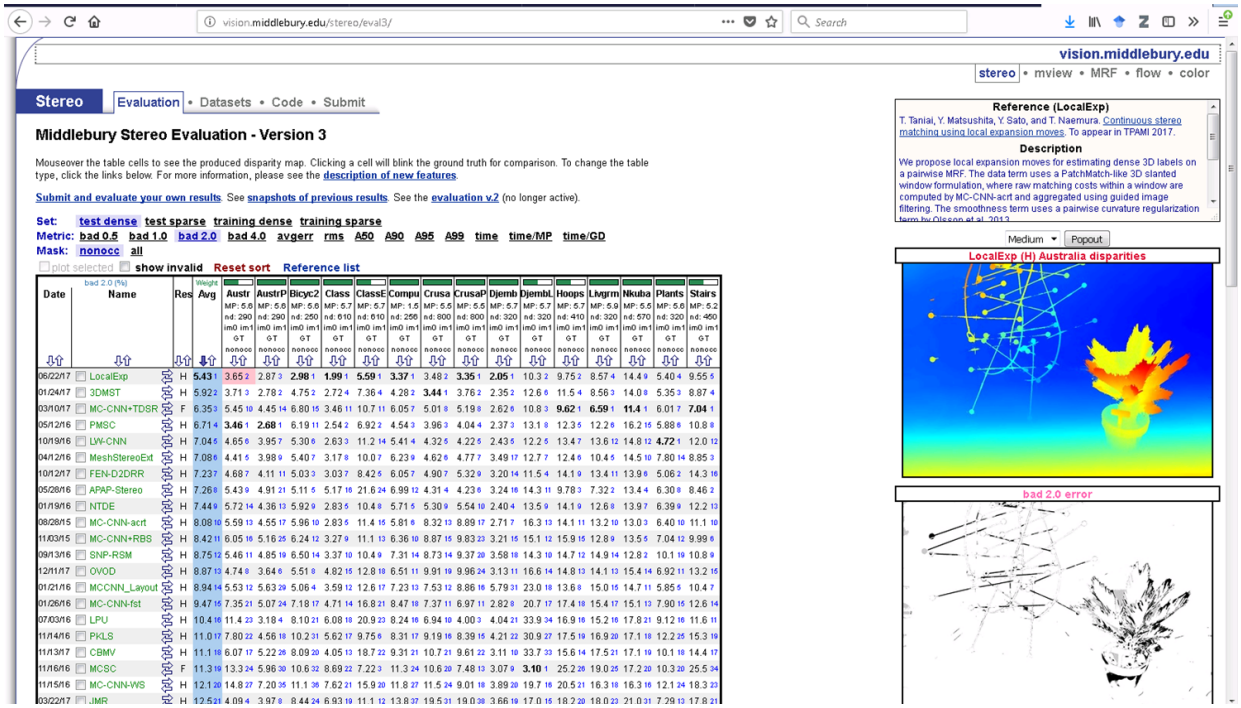


Fig. 1. Screenshot from the Middlebury Stereo Vision Benchmark page.

the field to easily select a promising algorithm for her or his application problem.

This is often due to a different perspective (identifying promising new research directions instead of just using existing algorithms), but often also due to lack of time, intense publication pressure and low standards.

The results are often highly unpleasant, though. Engineers with limited insight into optimization and even less time often resort to the first optimizer they find in the literature (or one that has been used for the specific problem domain before). There is a real danger that the best methods that the academic world has to offer do not make it into practical applications or only with long delays.

These problems are not limited to the domain of evolutionary algorithms but have appeared also in other fields. It is therefore highly inspirational to look how other fields have tried to solve the problem.

IV. Benchmark Suites in Computer Vision

In the field of computer vision there are a number of benchmarking frameworks, often found online and maintained by companies or universities. In the following we want to highlight two specific sites, the Middlebury stereo vision benchmark and the KITTI benchmark suite for autonomous driving related vision tasks. These benchmarks are standards in their respective fields, both for academic and industrial users and are widely used by authors and reviewers. They also substantially reduce the work researchers spent on testing, since all that is required is to download the data, process it and upload the algorithms' outputs to the websites. Statistical analysis, rankings, and visualization of results is done in a consistent way.

These two benchmarks will be shortly described below and then compared to benchmarking approaches in the Evolutionary Computing community.

A. Middlebury Stereo Vision Benchmark

The Middlebury Stereo Vision Benchmark [34] (*http://vision.middlebury.edu/stereo/*) is a standard benchmark for stereo camera image processing used basically for disparity (depth map) computation. A large number of algorithms (including published works and anonymous contributions) are compared according to a variety of performance indicators on several labeled test images. The website provides references to the tested algorithms, results on specific data sets (including a visualization of results) and extensive test statistics. Algorithms are ranked so that it is possible to clearly identify the currently best algorithm according to some performance indicator.

To give an estimate of the appeal of the webpage, Fig. 1 shows a screen shot of the webpage when a specific algorithm and test problem is selected (this affects the result visualization on the right side of the page). It is also possible to compare several algorithms directly against each other as shown in Fig. 2.

The authors of the webpage provide an evaluation SDK which can be used by the scientific community to benchmark their own algorithms and provide those results in a format that can be directly integrated into the main database and visualization.

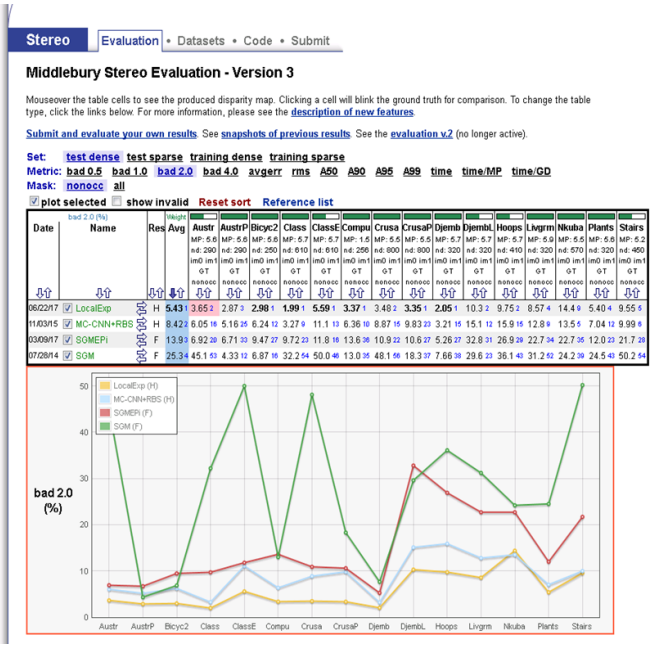


Fig. 2. Screenshot showing comparison of four methods on 15 test images (Middlebury).

Data is split into training and testing sets. Results on training data can be uploaded an unlimited number of times, but results on test data can only be submitted once per algorithm in order to avoid overfitting the algorithm to the test data. The necessity of this rule is a clear indicator that unscientific practices are a common problem in many domains.

The webpage is maintained by researchers from the Middlebury college with financial support from the NSF and Microsoft Research.

B. KITTI Automotive Vision Benchmark

While the Middlebury benchmark targets an academic audience, the KITTI vision benchmark [35] (*http://www.cvlibs.net/datasets/kitti/*) focuses more on industrial applications, especially for the automotive domain.

It is maintained by the Karlsruhe Institute of Technology (KIT) with financial support from Toyota Technological Institute at Chicago and also received test data contributions from other institutions including ourselves [36].

Test data is taken from real-world recordings in cars and evaluated for specific applications in the domain of Advanced Driver Assistance Systems (ADAS) and Autonomous Driving (AD).

There is test data for various computer vision applications, including stereo vision. KITTI is specifically targeting automotive audiences instead of the general computer vision community. In fact, as the webpage notes: "Preliminary experiments show that methods ranking high on established benchmarks such as Middlebury perform below average when being moved outside the laboratory to the real world". This shows that a proper definition of benchmark functions for the targeted audience is essential to provide the right guidelines to readers of benchmarks.

Similar to the Middlebury database, different algorithms are compared using a number of performance indicators. Again, a clear ranking is provided. Performance is evaluated using submitted algorithmic outputs (image processing results), but benchmark statistics are computed at the KITTI side. Correct results for the test suits are not published, in order to reduce cheating. A screenshot can be seen in Fig. 3.

For every contribution detailed test statistics are available including a visualization of performance, see Fig. 4.

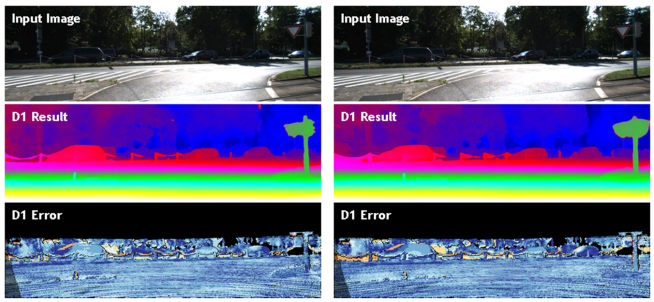


Fig. 4. Screenshot of performance visualization (KITTI).

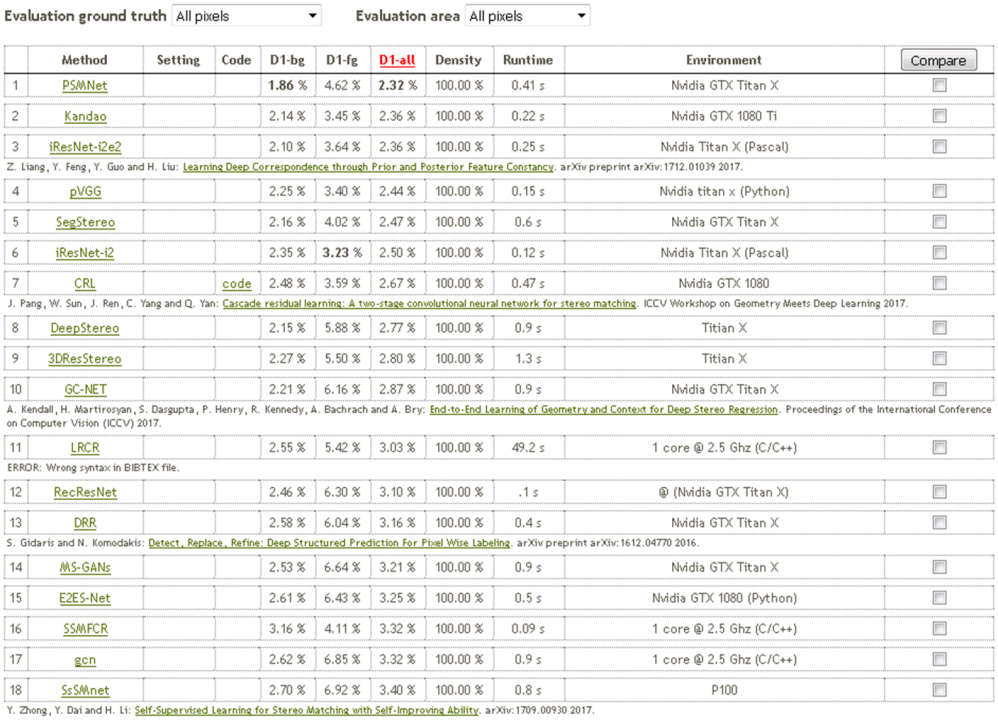
C. Comparison to evolutionary benchmarks

While in the field of evolutionary algorithms each new method brings its own set of benchmark results (with different performance metrics, competing algorithms, and benchmark suites), in at least subfields of computer vision there are centralized benchmark sites. Submission to those sites is a minimum requirement for all newly proposed algorithms to get published. New concepts are compared against all previous methods on all standard benchmarks and performance metrics. The long-term effort for the community is much lower, since every author only needs to test and document his own algorithm and there is no need to worry about the details of statistics. The computer vision community has encountered a number of problems in these benchmarks tests, like overfitting to test data, performance values too close to perfection to have any meaningful value or disputed performance metrics. Still the impact of these benchmarks is much higher than of benchmarks in our community. Since performance statistics are generated on the website using uploaded result data the extra workload for researchers is very low.

We believe that such a centralized benchmark repository could be used to substantially reduce benchmarking efforts for the academic community, reduce fraud and be far more useful for industry. The COCO benchmark suite [37] is a good step in this direction, but focuses on single- and bi-objective problems. It is also far from a standard for the community. Finally, results are rather difficult to interpret.

V. Summary: A Wish List from Industry

In this article we have outlined the practical challenges for selecting suitable optimization algorithms for industrial



Evaluation ground truth: All pixels Evaluation area: All pixels

	Method	Setting	Code	D1-bg	D1-fg	D1-all	Density	Runtime	Environment	Compare
1	PSMNet			**1.86 %**	4.62 %	**2.32 %**	100.00 %	0.41 s	Nvidia GTX Titan X	
2	Kandao			2.14 %	3.45 %	2.36 %	100.00 %	0.22 s	Nvidia GTX 1080 Ti	
3	iResNet-i2e2			2.10 %	3.64 %	2.36 %	100.00 %	0.25 s	Nvidia Titan X (Pascal)	
	Z. Liang, Y. Feng, Y. Guo and H. Liu: Learning Deep Correspondence through Prior and Posterior Feature Constancy. arXiv preprint arXiv:1712.01039 2017.									
4	pVGG			2.25 %	3.40 %	2.44 %	100.00 %	0.15 s	Nvidia titan x (Python)	
5	SegStereo			2.16 %	4.02 %	2.47 %	100.00 %	0.6 s	Nvidia GTX Titan X	
6	iResNet-i2			2.35 %	**3.23 %**	2.50 %	100.00 %	0.12 s	Nvidia Titan X (Pascal)	
7	CRL		code	2.48 %	3.59 %	2.67 %	100.00 %	0.47 s	Nvidia GTX 1080	
	J. Pang, W. Sun, J. Ren, C. Yang and Q. Yan: Cascade residual learning: A two-stage convolutional neural network for stereo matching. ICCV Workshop on Geometry Meets Deep Learning 2017.									
8	DeepStereo			2.15 %	5.88 %	2.77 %	100.00 %	0.9 s	Titian X	
9	3DResStereo			2.27 %	5.50 %	2.80 %	100.00 %	1.3 s	Titian X	
10	GC-NET			2.21 %	6.16 %	2.87 %	100.00 %	0.9 s	Nvidia GTX Titan X	
	A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach and A. Bry: End-to-End Learning of Geometry and Context for Deep Stereo Regression. Proceedings of the International Conference on Computer Vision (ICCV) 2017.									
11	LRCR			2.55 %	5.42 %	3.03 %	100.00 %	49.2 s	1 core @ 2.5 Ghz (C/C++)	
	ERROR: Wrong syntax in BIBTEX file.									
12	RecResNet			2.46 %	6.30 %	3.10 %	100.00 %	.1 s	@ (Nvidia GTX Titan X)	
13	DRR			2.58 %	6.04 %	3.16 %	100.00 %	0.4 s	Nvidia GTX Titan X	
	S. Gidaris and N. Komodakis: Detect, Replace, Refine: Deep Structured Prediction For Pixel Wise Labeling. arXiv preprint arXiv:1612.04770 2016.									
14	MS-GANs			2.53 %	6.64 %	3.21 %	100.00 %	0.9 s	Nvidia GTX Titan X	
15	E2ES-Net			2.61 %	6.43 %	3.25 %	100.00 %	0.5 s	Nvidia GTX 1080 (Python)	
16	SSMFCR			3.16 %	4.11 %	3.32 %	100.00 %	0.09 s	1 core @ 2.5 Ghz (C/C++)	
17	gcn			2.62 %	6.85 %	3.32 %	100.00 %	0.9 s	1 core @ 2.5 Ghz (C/C++)	
18	SsSMnet			2.70 %	6.92 %	3.40 %	100.00 %	0.8 s	P100	
	Y. Zhong, Y. Dai and H. Li: Self-Supervised Learning for Stereo Matching with Self-Improving Ability. arXiv:1709.00930 2017.									

Fig. 3. Screenshot from the KITTI automotive vision benchmark page.

applications. We tried to sketch some of the short-comings of current benchmarking practices in academia and introduced two examples for benchmarking suites from a different domain, computer vision, which might trigger some ideas for similar approaches in evolutionary algorithms.

Finally, we would like to propose a solution in the form of a requirement list. This should be taken as a basis for discussion. We of course know that there is no perfect ranking and a-priori, i.e. without knowing the structure of the optimization problem, it is impossible to identify the best suited optimizer. Still we should target to maximize the probability that an engineer, who is not an expert in the field, would finally decide on a proper approach, after spending a moderate amount of time to select the algorithm. The practical relevance of a benchmark is therefore to help the practitioner pick a suitable optimizer.

- Provide a single site for benchmarking, this will greatly enhance acceptance
- A large and diverse set of academic and real-world test functions should be used.
- Various performance indicators (hypervolume, IGD, number of solutions dominating a baseline solution, chance of not finding any valid solution, etc.) should be offered.
- Encourage the authors to provide computational performance data, like runtimes and used computing equipment.
- Benchmark statistics are automatically generated by the site, not the authors.
- Clearly show an explicit ranking of algorithms. Yes, it is unfair at times, but will generate the necessary incentives for researchers to improve their methods and allows a simple assessment of algorithmic quality.
- On demand, detailed statistics and performance on specific test functions needs to be shown. Weighting factors for the final ranking should be flexible. All test data should be available for download (except for the optimal parameter values of the solutions).
- Consider methods to reduce spamming and various frauds. As soon as the benchmark becomes really important some people will try to cheat.
- Encourage open source software in common standards. This will increase the chances of industry picking up the algorithm and of finding bugs (which normally reduce performance).
- Request detailed information about parametrization of algorithms (population size, mutation rates, crossover probabilities, etc.). Encourage authors to provide information on performance for other values of these parameters.
- Synchronize benchmarking efforts with optimization tool vendors. The best algorithms should quickly appear in the most relevant tools if they provide a clear benefit.
- Consider test situations that are actually needed in industry. This requires a careful look at benchmark functions, computing budgets, performance indicators and MCDM. For example, the preference-based optimization is very promising for industrial applications but so far very little covered in benchmark tests.
- Make a good performance on the benchmark site tests one of the most (but not the only) important criteria for paper acceptance.

- The webpage should be intuitive to understand, the most important ranking should be the default view on the data.

Many (maybe all) of the above points can be disputed. They are probably one-sided and difficult to implement. However, it is our hope that they provide a better understanding on what is important for experts from different fields to use our algorithms.

ACKNOWLEDGMENTS

We thank Nils Einecke for providing valuable insights on the Middlebury and KITTI benchmarks.

REFERENCES

[1] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *CoRR*, vol. abs/1701.00879, 2017. [Online]. Available: http://arxiv.org/abs/1701.00879

[2] G. Endicott, M. Olhofer, T. Arima, and T. Sonoda, "A novel transonic fan swept outlet guide vane using 3d design optimization," in *Proceedings of the ASME Turbo Expo (ASME)*. American Society of Mechanical Engineers, June 2014.

[3] K. Foli, T. Okabe, M. Olhofer, Y. Jin, and B. Sendhoff, "Optimization of micro heat exchanger: CFD, analytical approach and multi-objective evolutionary algorithms," *International Journal of Heat and Mass Transfer*, vol. 49, pp. 1090–1099, 2006.

[4] N. Aulig and M. Olhofer, "State-based representation for structural topology optimization and application to crashworthiness," in *Proceedings of: IEEE Congress on Evolutionary Computation*. IEEE, July 2016.

[5] T. Rodemann, K. Narukawa, M. Fischer, and M. Awada, "Many-objective optimization of a hybrid car controller," in *Applications of Evolutionary Computation*, ser. Lecture Notes in Computer Science, A. M. Mora and G. Squillero, Eds., vol. 9028. Springer International Publishing, 2015, pp. 593–603.

[6] N. Hansen and S. Kern, "Evaluating the CMA evolution strategy on multimodal test functions." in *Eighth International Conference on Parallel Problem Solving from Nature PPSN VIII*. Berlin: Springer, 2004, pp. 282–291.

[7] S. van Rijn, H. Wang, M. van Leeuwen, and T. Bäck, "Evolving the structure of evolution strategies," *CoRR*, vol. abs/1610.05231, 2016. [Online]. Available: http://arxiv.org/abs/1610.05231

[8] L. Bezerra, M. López-Ibánez, and T. Stützle, "A large-scale experimental evaluation of high-performing multi-and many-objective evolutionary algorithms," IRIDIA, Tech. Rep. 2017-005, 2017.

[9] P. J. Fleming, R. C. Purshouse, and R. J. Lygoe, "Many-objective optimization: An engineering design perspective," in *International conference on evolutionary multi-criterion optimization*. Springer, 2005, pp. 14–32.

[10] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, pp. 1653–1669, 2007.

[11] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," ETH Zurich, Tech. Rep., 2008. [Online]. Available: http://www.johannesbader.ch/publications.html

[12] H. Sato, H. Aguirre, and K. Tanaka, "Controlling dominance area of solutions and its impact on the performance of MOEAs," in *Evolutionary Multi-Criterion Optimization: EMO 2007*, ser. Lecture Notes in Computer Science, S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, Eds., vol. 4403. Springer, 2007, pp. 5–20.

[13] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[14] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, 2016.

[15] K. Deb and H. Jain, "Handling many-objective problems using an improved NSGA-II procedure," in *Proc. of 2012 IEEE Congress on Evolutionary Computation (World Congress on Computational Intelligence)*. IEEE, 2012, pp. 1–8.

[16] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and Y. Jin, "Evolutionary many-objective optimization of hybrid electric vehicle control: from general optimization to preference articulation," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 1, no. 2, pp. 97–111, February 2017.

[17] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: A survey," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 13, 2015.

[18] K. Miettinen, "Survey of methods to visualize alternatives in multiple criteria decision making problems," *OR Spectrum*, vol. 36, no. 1, pp. 3–37, 2014. [Online]. Available: http://dx.doi.org/10.1007/s00291-012-0297-0

[19] T. Wagner and H. Trautmann, "Integration of preferences in hypervolume-based multiobjective evolutionary algorithms by means of desirability functions," *Evolutionary Computation, IEEE Transactions on*, vol. 14, no. 5, pp. 688–701, 2010.

[20] Y. Ogino, R. Iida, and T. Rodemann, "Using desirability functions for many-objective optimization of a hybrid car controller," in *GECCO 2017 conference companion*, 2017.

[21] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 1, may 2002, pp. 825 –830.

[22] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[23] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[24] J. Lehman and K. O. Stanley, "Abandoning objectives: Evolution through the search for novelty alone," *Evolutionary Computation*, vol. 19, no. 2, pp. 189–223, 2011, pMID: 20868264.

[25] M. López-Ibáñez, J. Dubois-Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle, "The irace package: Iterated racing for automatic algorithm configuration," *Operations Research Perspectives*, vol. 3, pp. 43–58, 2016.

[26] R. A. Krohling, R. Lourenzutti, and M. Campos, "Ranking and comparing evolutionary algorithms with hellinger-topsis," *Appl. Soft Comput.*, vol. 37, pp. 217–226, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.asoc.2015.08.012

[27] D. Hadka and P. Reed, "Borg: An auto-adaptive many-objective evolutionary computing framework," *Evolutionary computation*, vol. 21, no. 2, pp. 231–259, 2013.

[28] K. Sörensen, "Metaheuristics-the metaphor exposed," *International Transactions in Operational Research*, vol. 22, no. 1, pp. 3–18, 2015.

[29] A.-T. Nguyen, S. Reiter, and P. Rigo, "A review on simulation-based optimization methods applied to building performance analysis," *Applied Energy*, vol. 113, pp. 1043–1058, 2014.

[30] esteco, "modefrontier," 2017. [Online]. Available: www.esteco.com/modefrontier

[31] OptiY, 2017. [Online]. Available: www.optiY.de

[32] Mathworks, "Matlab global optimization toolbox," 2017. [Online]. Available: https://www.mathworks.com/products/global-optimization.html

[33] DATADVANCE, 2017. [Online]. Available: https://www.datadvance.net/product/pseven/

[34] D. Scharstein and R. Szeliski, "Middlebury stereo evaluation-version 2," http://vision.middlebury.edu/stereo/eval, 2002.

[35] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3354–3361.

[36] J. Fritsch, T. Kühnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE, 2013, pp. 1693–1700.

[37] (2017) Coco (comparing continuous optimisers). [Online]. Available: http://coco.gforge.inria.fr/doku.php?id=start