A Brief Introduction to (Evolutionary) Multiobjective Optimization

Dimo Brockhoff
dimo.brockhoff@inria.fr
http://researchers.lille.inria.fr/~brockhof/
A Brief Introduction to (Evolutionary) Multiobjective Optimization

Dimo Brockhoff
dimo.brockhoff@inria.fr
http://researchers.lille.inria.fr/~brockhof/

+ (even shorter)
Intro to Coco Framework
A Brief Introduction to Multiobjective Optimization

**Multiobjective Optimization:**
problems where multiple objectives have to be optimized simultaneously
Observations: ① there is no single optimal solution, but
② some solutions (●) are better than others (○)
A Brief Introduction to Multiobjective Optimization

Observations: ① there is no single optimal solution, but
② some solutions (●) are better than others (○)

Vilfredo Pareto
(1848–1923)
wikipedia
Observations: 1. there is no single optimal solution, but 2. some solutions (●) are better than others (○).

Pareto front (Pareto efficient frontier)

Vilfredo Pareto (1848 – 1923)

wikipedia
Observations: 1. there is no single optimal solution, but 2. some solutions (●) are better than others (○)

- decision making
  - selecting a solution
- optimization
  - finding the good solutions

performance

cost
Selecting a Solution: Examples

Possible Approaches:

1. **ranking**: performance more important than cost
Selecting a Solution: Examples

Possible Approaches:

1. ranking: performance more important than cost
2. constraints: cost must not exceed 2400

Graph showing performance and cost with points above 2400 marked as too expensive.
When to Make the Decision

Before Optimization:

- rank objectives,
- define constraints,…
- search for one (good) solution
When to Make the Decision

Before Optimization:

- rank objectives, define constraints,…
- search for one (good) solution

![Graph showing performance vs. cost with decision points marked]

When to Make the Decision

- Cost is too expensive for certain points on the graph.
When to Make the Decision

**Before Optimization:**

- rank objectives, define constraints,…
- search for one (good) solution

**After Optimization:**

- search for a set of (good) solutions
- select one solution considering constraints, etc.
When to Make the Decision

Before Optimization:

- rank objectives, define constraints,…
- search for one (good) solution

After Optimization:

- search for a set of (good) solutions
- select one solution considering constraints, etc.

Focus: learning about a problem
- trade-off surface
- interactions among criteria
- structural information
- also: interactive optimization
Two Communities...

- beginning in 1950s/1960s
- bi-annual conferences since 1975
- background in economics, math, management science
- both optimization and decision making

- quite young field (first papers in mid 1980s)
- bi-annual conference since 2001
- background evolutionary computation (applied math, computer science, engineering, ...)
- focus on optimization algorithms
MCDM track at EMO conference since 2009
special sessions on EMO at the MCDM conference since 2008
joint Dagstuhl seminars since 2004
One of the Main Differences

Blackbox optimization

\[ x \in S \xrightarrow{f} (f_1(x), \ldots, f_k(x)) \]

only mild assumptions

\[ \rightarrow \text{EMO therefore well-suited for real-world (engineering) problems} \]

- non-linear
- noisy
- uncertain
- non-differentiable
- expensive (integrated simulations)
- many objectives
- huge search spaces
- many constraints

© Dimo Brockhoff, INRIA Lille – Nord Europe
Evolutionary Multiobjective Optimization

- set-based algorithms
- therefore possible to approximate the Pareto front in one run

The Other Main Difference

environmental selection

mating selection

evaluation

variation

Pareto front approximation

performance

cost
The Big Picture

Basic Principles of Multiobjective Optimization
- algorithm design principles and concepts
- performance assessment & benchmarking
Differences To Single-Objective Problems

- **Single objective**
  - Solutions always comparable
  - Clearly defined optimum
  - With \( a \) better than \( b \)
    - If \( f(a) \leq f(b) \)

- **Multiple objectives**
  - Solutions not always comparable
  - Set of Pareto-optimal solutions
  - With \( a \) better than \( b \)
    - If \( f(a) \) prefrel \( f(b) \)

Even more complicated: sought are sets!
Most Common Example: Pareto Dominance

\[ u \text{ weakly Pareto dominates } v \ (u \leq_{\text{par}} v) : \ \forall 1 \leq i \leq k : f_i(u) \leq f_i(v) \]

\[ u \text{ Pareto dominates } v \ (u <_{\text{par}} v) : \ u \leq_{\text{par}} v \ \land \ v \not\leq_{\text{par}} u \]
Different Notions of Dominance

- \( \varepsilon \)-dominance
- Pareto dominance
- Cone dominance
The minimal set of a preordered set \((Y, \leq)\) is defined as

\[
\text{Min}(Y, \leq) := \{ a \in Y | \forall b \in Y : b \leq a \Rightarrow a \leq b \}
\]

Pareto-optimal set \(\text{Min}(S, \leq_{\text{par}})\)

non-optimal decision vector

Pareto-optimal front

non-optimal objective vector
A multiobjective problem is as such underspecified …because not any Pareto-optimum is equally suited!

Additional preferences are needed to tackle the problem:

**Solution-Oriented Problem Transformation:**
Classical approach: Induce a total order on the decision space, e.g., by aggregation

**Set-Oriented Problem Transformation:**
Recent view on EMO: First transform problem into a set problem and then define an objective function on sets [Zitzler et al. 2010]

Preferences are needed in both cases, but the latter are weaker!
Solution-based Approaches (classical approaches)
A scalarizing function $s$ is a function $s : Z \rightarrow \mathbb{R}$ that maps each objective vector $u = (u_1, \ldots, u_n) \in Z$ to a real value $s(u) \in \mathbb{R}$.
Solution-Oriented Problem Transformations

Example 1: weighted sum approach

\[ y = w_1 y_1 + \ldots + w_k y_k \]

Disadvantage: not all Pareto-optimal solutions can be found if the front is not convex
Solution-Oriented Problem Transformations

### Example 2: weighted Tchebycheff

\[ y = \max | \lambda_i(u_i - z_i) | \]

Several other scalarizing functions are known, see e.g. [Miettinen 1999]
Set-based Approaches
(Evolutionary Multiobjective Optimizers)
Algorithm Design: Particular Aspects

representation

fitness assignment

mating selection

parameters

environmental selection

variation operators
General Scheme of Most Dominance-Based EMO

- **mating selection** (stochastic)
- **environmental selection** (greedy heuristic)
- **fitness assignment**
  - partitioning into dominance classes
  - rank refinement within dominance classes
Ranking of the Population Using Dominance

... is based on pairwise comparisons of the individuals only

[Goldberg 1989]

Examples:

- Dominance rank
- Dominance depth

...
Goal: rank incomparable solutions within a dominance class

1. Density information (good for search, but usually incompatible with Pareto-dominance)

2. Quality indicator (good for set quality)
Density Estimation

crowding distance:

- sort solutions wrt. each objective

- crowding distance to neighbors:

\[ d(i) = \sum_{\text{obj. } m} |f_m(i - 1) - f_m(i + 1)| \]
Selection in SPEA2 and NSGA-II can result in *deteriorative* cycles

non-dominated solutions already found can be lost
Hypervolume-Based Selection

Latest Approach (SMS-EMOA, MO-CMA-ES, HypE, …)
use hypervolume indicator to guide the search

Main idea
Delete solutions with the smallest hypervolume loss
d(s) = \( I_H(P) - I_H(P / \{s\}) \)
iteratively

But: can also result in cycles if reference point is not constant [Judt et al. 2011]
and is expensive to compute exactly [Bringmann and Friedrich 2009]

Moreover: HypE [Bader and Zitzler 2011]
Sampling + Contribution if more than 1 solution deleted
Example Algorithm: the MO-CMA-ES

CMA-ES [remember talk of Rodolphe]

- Covariance Matrix Adaptation Evolution Strategy
  [Hansen and Ostermeier 1996, 2001]
- the state-of-the-art numerical black box optimizer for large budgets and difficult functions [Hansen et al. 2010]

CMA-ES for multiobjective optimization

- “THE” MO-CMA-ES does not exist
- original one of [Igel et al. 2007] in ECJ
- improved success definition [Voß et al. 2010] at GECCO 2010
- all based on combination of $\mu$ single (1+1)-CMA-ES
(1+λ)-CMA-ES

\[ a^{(g)} = (x^{(g)}, p_{\text{succ}}^{(g)}, \sigma^{(g)}, p_c^{(g)}, C^{(g)}) \]

Algorithm 1: (1+λ)-CMA-ES

1. \( g = 0 \), initialize \( a_{\text{parent}}^{(g)} \)
2. repeat
   3. \( a_{\text{parent}}^{(g+1)} \leftarrow a_{\text{parent}}^{(g)} \)
   4. for \( k = 1, \ldots, \lambda \) do
      5. \( x_k^{(g+1)} \sim \mathcal{N}(x_{\text{parent}}^{(g)}, \sigma^{(g)}_2 C^{(g)}) \)
      6. updateStepsize \( \left( a_{\text{parent}}^{(g+1)}, \frac{\lambda_{\text{succ}}^{(g+1)}}{\lambda} \right) \)
      7. if \( f(x_{1:\lambda}^{(g+1)}) \leq f(x_{\text{parent}}^{(g)}) \) then
         8. \( x_{\text{parent}}^{(g+1)} \leftarrow x_{1:\lambda}^{(g+1)} \)
      9. updateCovariance \( \left( a_{\text{parent}}^{(g+1)}, \frac{x_{\text{parent}}^{(g+1)} - x_{\text{parent}}^{(g)}}{\sigma_{\text{parent}}^{(g)}} \right) \)
   10. \( g \leftarrow g + 1 \)
3. until stopping criterion is met
(1+λ)-CMA-ES: Updates

Procedure `updateStepsize` \( a = [x, \overline{p}_{\text{succ}}, \sigma, p_c, C], p_{\text{succ}} \) 

1. \( \overline{p}_{\text{succ}} \leftarrow (1 - c_p) \overline{p}_{\text{succ}} + c_p p_{\text{succ}} \)
2. \( \sigma \leftarrow \sigma \cdot \exp \left( \frac{1}{d} \frac{\overline{p}_{\text{succ}} - p_{\text{succ}}^{\text{target}}}{1 - p_{\text{succ}}^{\text{target}}} \right) \)

Procedure `updateCovariance` \( a = [x, \overline{p}_{\text{succ}}, \sigma, p_c, C], x_{\text{step}} \in \mathbb{R}^n \) 

1. if \( \overline{p}_{\text{succ}} < p_{\text{thresh}} \) then
   2. \( p_c \leftarrow (1 - c_c) p_c + \sqrt{c_c (2 - c_c)} x_{\text{step}} \)
   3. \( C \leftarrow (1 - c_{\text{cov}}) C + c_{\text{cov}} \cdot p_c p_c^T \)
2. else
   5. \( p_c \leftarrow (1 - c_c) p_c \)
   6. \( C \leftarrow (1 - c_{\text{cov}}) C + c_{\text{cov}} \cdot (p_c p_c^T + c_c (2 - c_c) C) \)
Concrete MO-CMA-ES Baseline Algorithm

\[ \mu \times (1+1)\text{-CMA-ES: } a_i^{(g)} = (x_i^{(g)}, p_{\text{succ},i}^{(g)}, \sigma_i^{(g)}, p_{c,i}^{(g)}, C_i^{(g)}) \]
Concrete MO-CMA-ES Baseline Algorithm

\[ \mu \times (1+1)\text{-CMA-ES}: \quad a_i^{(g)} = (x_i^{(g)}, \bar{p}_{\text{succ},i}^{(g)}, \sigma_i^{(g)}, p_{c,i}^{(g)}, C_i^{(g)}) \]
Concrete MO-CMA-ES Baseline Algorithm

\[ a_i^{(g)} = (x_i^{(g)}, \overline{p}_{\text{succ},i}^{(g)}, \sigma_i^{(g)}, p_{c,i}^{(g)}, C_i^{(g)}) \]
Concrete MO-CMA-ES Baseline Algorithm

\[ a_i^{(g)} = (x_i^{(g)}, \bar{p}_{\text{succ},i}, \sigma_i^{(g)}, p_{c,i}^{(g)}, C_i^{(g)}) \]
Concrete MO-CMA-ES Baseline Algorithm

$$\mu \times (1+1)\text{-CMA-ES: } a_i^{(g)} = (x_i^{(g)}, \bar{p}_{\text{succ},i}^{(g)}, \sigma_i^{(g)}, p_{c,i}^{(g)}, C_i^{(g)})$$
MO-CMA-ES baseline algorithm

- $\mu \times (1+1)$-CMA-ES
- hypervolume-based selection
- update of CMA strategy parameters based on different success notions

Success Definitions:

- original success [Igel et al. 2007]: if offspring dominates parent
- improved success [Voß et al. 2010]: if offspring selected into new population

Available Implementations:

- Baseline in Shark machine learning library (C++)
  - http://image.diku.dk/shark/
- Now also available in MATLAB
  - easy prototyping of new ideas
  - visualization of algorithm’s state variables (similar to CMA-ES)
The Big Picture

Basic Principles of Multiobjective Optimization

- algorithm design principles and concepts
- performance assessment & benchmarking
Why Benchmarking Algorithms?

- **Understanding** of algorithms
- **Algorithm selection**
- Putting algorithms to a **standardized test**
  - simplify judgement
  - simplify comparison
  - regression test under algorithm changes

We can measure performance on

- **real world problems**
  - expensive, often limited to certain domain
- "artificial" benchmark functions
  - cheap
  - controlled
  - data acquisition is comparatively easy
  - problem of representativity
Once Upon a Time...

... multiobjective EAs were mainly compared visually:

ZDT6 benchmark problem: IBEA, SPEA2, NSGA-II
Two Approaches for Empirical Studies

Attainment function approach:
- Applies statistical tests directly to the samples of approximation sets
- Gives detailed information about how and where performance differences occur

Quality indicator approach:
- First, reduces each approximation set to a single value of quality
- Applies statistical tests to the samples of quality values

see e.g. [Zitzler et al. 2003]
Attainment Plots

50% attainment surface for IBEA, SPEA2, NSGA2 (ZDT6)

latest implementation online at
http://eden.dei.uc.pt/~cmfonsec/software.html
see [Fonseca et al. 2011]
**Comparison method** \( C = \text{quality measure(s)} + \text{Boolean function} \)

**Goal:** compare two Pareto set approximations \( A \) and \( B \)

<table>
<thead>
<tr>
<th>Quality Indicator Approach</th>
<th>( A )</th>
<th>( B )</th>
</tr>
</thead>
<tbody>
<tr>
<td>hypervolume</td>
<td>432.34</td>
<td>420.13</td>
</tr>
<tr>
<td>distance</td>
<td>0.3308</td>
<td>0.4532</td>
</tr>
<tr>
<td>diversity</td>
<td>0.3637</td>
<td>0.3463</td>
</tr>
<tr>
<td>spread</td>
<td>0.3622</td>
<td>0.3601</td>
</tr>
<tr>
<td>cardinality</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

→ “\( A \) better”
Example: Box Plots

epsilon indicator  hypervolume  R indicator

IBEA NSGA-II SPEA2  IBEA NSGA-II SPEA2  IBEA NSGA-II SPEA2

DTLZ2

IBEA NSGA-II SPEA2  IBEA NSGA-II SPEA2  IBEA NSGA-II SPEA2

Knapsack

ZDT6

remark: not all quality indicators comply with the Pareto dominance hypervolume and \( \epsilon \)-indicator a good choice
small excursion:

**Single-objective Blackbox Optimization Benchmarking in Practice (using COCO)**

COCO (COmparing Continuous Optimizers): a tool for black-box optimization benchmarking

[slides borrowed from Nikolaus Hansen]
This is the COCO download page.

Last release: **30/05/2012** v11.06

**BBOB (5MB)** is all that is needed to run the benchmarking experiments and compile a template paper (gathering post-processed results).

**BBOB (35MB)** contains all files, as listed below.

- **CODE:**
  - tar code in Matlab/Octave to run experiments
  - tar code in C to run experiments
  - tar code in Java to run experiments
  - tar code in Python to run experiments and post-processing and latex templates (3MB)
  - tar R package to run experiments

- **DOCS:**
  - pdf description of experimental procedure
  - pdf (12MB) noiseless functions documentation with figures
  - pdf noiseless functions documentation, version without figures
  - pdf (19MB) noisy function documentation with figures
  - pdf noisy function documentation, version without figures
  - pdf software user documentation
  - html online post-processing package documentation

BUGS for older versions:

- Bugs in version 11.05:
## BBOB in practice

<table>
<thead>
<tr>
<th>Name</th>
<th>Date Modified</th>
<th>Size</th>
<th>Kind</th>
</tr>
</thead>
<tbody>
<tr>
<td>bbob.v11.06</td>
<td>Today, 1:15</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>c</td>
<td>May 30, 2012 12:07</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>docs</td>
<td>October 27, 2012 0:58</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>java</td>
<td>May 30, 2012 12:07</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>latextemplates</td>
<td>May 30, 2012 12:06</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>matlab</td>
<td>May 30, 2012 12:06</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>python</td>
<td>May 30, 2012 12:06</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>r</td>
<td>May 30, 2012 12:07</td>
<td>--</td>
<td>Folder</td>
</tr>
</tbody>
</table>

[Image of file tree]
# BBOB in practice

<table>
<thead>
<tr>
<th>Name</th>
<th>Date Modified</th>
<th>Size</th>
<th>Kind</th>
</tr>
</thead>
<tbody>
<tr>
<td>bbob.v11.06</td>
<td>Today, 1:15</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>c</td>
<td>May 30, 2012 12:07</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>docs</td>
<td>October 27, 2012 0:58</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>java</td>
<td>May 30, 2012 12:07</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>latextemplates</td>
<td>May 30, 2012 12:06</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>matlab</td>
<td>May 30, 2012 12:06</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>benchmarkinfos.txt</td>
<td>February 9, 2009 16:29</td>
<td>4 KB</td>
<td>Gedit Document</td>
</tr>
<tr>
<td>benchmarks.m</td>
<td>February 10, 2011 16:24</td>
<td>86 KB</td>
<td>Object File</td>
</tr>
<tr>
<td>benchmarksnoisy.m</td>
<td>February 10, 2011 16:24</td>
<td>102 KB</td>
<td>Object File</td>
</tr>
<tr>
<td>exampleexperiment.m</td>
<td>February 1, 2012 19:52</td>
<td>4 KB</td>
<td>Object File</td>
</tr>
<tr>
<td>exampletiming.m</td>
<td>March 7, 2012 14:40</td>
<td>4 KB</td>
<td>Object File</td>
</tr>
<tr>
<td>fgeneric.m</td>
<td>December 7, 2011 17:56</td>
<td>33 KB</td>
<td>Object File</td>
</tr>
<tr>
<td>LICENSE.txt</td>
<td>February 1, 2012 20:23</td>
<td>4 KB</td>
<td>Gedit Document</td>
</tr>
<tr>
<td>MY_OPTIMIZER.m</td>
<td>February 14, 2011 19:30</td>
<td>4 KB</td>
<td>Object File</td>
</tr>
<tr>
<td>README.txt</td>
<td>May 19, 2011 10:47</td>
<td>4 KB</td>
<td>Gedit Document</td>
</tr>
<tr>
<td>python</td>
<td>May 30, 2012 12:06</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>r</td>
<td>May 30, 2012 12:07</td>
<td>--</td>
<td>Folder</td>
</tr>
</tbody>
</table>

© Dimo Brockhoff, INRIA Lille – Nord Europe

EMO tutorial, La Rochelle, November 5, 2014
Matlab script (exampleexperiment.m):

dimensions = [2, 3, 5, 10, 20, 40]; % small dimensions first, for CPU reasons
functions = benchmarks('FunctionIndices'); % or benchmarksnoisy(...)
instances = [1:5, 31:40]; % 15 function instances

for dim = dimensions
    for ifun = functions
        for iinstance = instances
            fgeneric('initialize', ifun, iinstance, datapath, opt); %
            MY_OPTIMIZER('fgeneric', dim, fgeneric('ftarget'), eval(maxfunevals) - fg
            disp(sprintf([' f%d in %d-D, instance %d: FEs=%d with %d restarts, fbest
            fgeneric('finalize'); %
            end
            disp([' date and time: ' num2str(clock, ' %.0f')]));
            end
            disp(sprintf('---- dimension %d-D done ----', dim));
        end
    end
end

Interface: MY_OPTIMIZER(function_name, dimension, optional_args)
Running the experiment at an OS shell:

```
$ nohup nice octave < exampleexperiment.m > output.txt &
$ less output.txt
```

GNU Octave, version 3.6.3
Copyright (C) 2012 John W. Eaton and others.
This is free software; see the source code for copying conditions.
[...]
Read http://www.octave.org/bugs.html to learn how to submit bug reports.

For information about changes from previous versions, type `news'.

```text
f1 in 2-D, instance 1: FE=242, fbest-ftarget=-8.1485e-10, elapsed time [h]: 0.00
f1 in 2-D, instance 2: FE=278, fbest-ftarget=-6.0931e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 3: FE=242, fbest-ftarget=-9.2281e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 4: FE=302, fbest-ftarget=-4.5997e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 5: FE=230, fbest-ftarget=-9.8350e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 6: FE=284, fbest-ftarget=-7.0829e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 7: FE=278, fbest-ftarget=-6.5999e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 8: FE=272, fbest-ftarget=-8.7044e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 9: FE=248, fbest-ftarget=-2.6316e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 10: FE=302, fbest-ftarget=-4.6779e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 11: FE=272, fbest-ftarget=-5.1499e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 12: FE=260, fbest-ftarget=-8.8635e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 13: FE=266, fbest-ftarget=-2.5484e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 14: FE=218, fbest-ftarget=-9.9961e-09, elapsed time [h]: 0.00
f1 in 2-D, instance 15: FE=248, fbest-ftarget=-7.5842e-09, elapsed time [h]: 0.00

date and time: 2013 3 29 19 59 26
f2 in 2-D, instance 1: FE=824, fbest-ftarget=-7.0206e-09, elapsed time [h]: 0.00
f2 in 2-D, instance 2: FE=572, fbest-ftarget=-9.2822e-09, elapsed time [h]: 0.00
```

BBOB in practice
## BBOB in practice

<table>
<thead>
<tr>
<th>Name</th>
<th>Date Modified</th>
<th>Size</th>
<th>Kind</th>
</tr>
</thead>
<tbody>
<tr>
<td>bbob.v13.05</td>
<td>March 8, 2013 13:04</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>c</td>
<td>March 5, 2013 23:04</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>docs</td>
<td>March 6, 2013 13:56</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>java</td>
<td>March 5, 2013 23:04</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>latextemplates</td>
<td>March 5, 2013 23:04</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>matlab</td>
<td>March 5, 2013 23:02</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>python</td>
<td>Today, 20:08</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>bbob_pproc</td>
<td>March 5, 2013 23:03</td>
<td>--</td>
<td>Folder</td>
</tr>
<tr>
<td>bbobbenchmarks.py</td>
<td>November 12, 2012 16:56</td>
<td>74 KB</td>
<td>Python script</td>
</tr>
<tr>
<td>benchmarkinfos.txt</td>
<td>February 9, 2009 16:29</td>
<td>4 KB</td>
<td>Gedit ...ument</td>
</tr>
<tr>
<td>exampleexperiment.py</td>
<td>February 22, 2013 14:26</td>
<td>4 KB</td>
<td>Python script</td>
</tr>
<tr>
<td>exampletiming.py</td>
<td>November 12, 2012 16:56</td>
<td>4 KB</td>
<td>Python script</td>
</tr>
<tr>
<td>fgeneric.py</td>
<td>March 3, 2013 19:33</td>
<td>25 KB</td>
<td>Python script</td>
</tr>
<tr>
<td>LICENSE.txt</td>
<td>February 1, 2012 20:23</td>
<td>4 KB</td>
<td>Gedit ...ument</td>
</tr>
<tr>
<td>README.txt</td>
<td>November 12, 2012 16:56</td>
<td>4 KB</td>
<td>Gedit ...ument</td>
</tr>
<tr>
<td>r</td>
<td>March 5, 2013 23:05</td>
<td>--</td>
<td>Folder</td>
</tr>
</tbody>
</table>
Post-processing at the OS shell:

$ python codepath/bbob_pproc/rungeneric.py datopath

[...]

$ pdflatex templateACMarticle.tex

[...]
Black-Box Optimization Benchmarking Template for Noiseless Function Testbed

Draft version *
Forename Name

ABSTRACT

Categories and Subject Descriptors
G.1.0 [Numerical Analysis]: Optimization; global optimization, unconstrained optimization; F.2.1 [Analysis of Algorithms and Problem Complexity]: Numerical Algorithms and Problems

General Terms
Algorithm

Keywords
Heterogeneous, Black-box optimization, Evolutionary computation

1. RESULTS

Results from experiments according to [7] on the benchmark functions given in [7, 7] are presented in Figures 1 and 2 and in Table 1.

*Camera-ready paper due April 17th.

Permission to make digital or hard copies of all or part of the work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GECCO '99, July 8-12, 2009, Montreal Quebec, Canada.
Copyright 2009 ACM 1-58113-4555-6/09/00.

Figure 1: Expected Running Time (ERT; ●) to reach $f_{max} + \Delta f$ and median number of function evaluations of successful trials (○), shown for $\Delta f = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$ (the exponent is given in the legend of $\Delta f$ and $f_{max}$) versus dimension in log-log presentation. The ERT($\Delta f$) equals to $nFE$($\Delta f$) divided by the number of successful trials, where a trial is successful if $f_{max} + \Delta f$ was surpassed during the trial. The $nFE$($\Delta f$) are the total number of function evaluations while $f_{max} + \Delta f$ was not surpassed during the trial from all respective trials (successful and unsuccessful), and $f_{max}$ denotes the optimal function value. Crosses (×) indicate the total number of function evaluations ($nFE$($\Delta f$)). Numbers above ERT-symbols indicate the number of successful trials. Annotated numbers on the ordinate are decimal logarithms. Additional grid lines show linear and quadratic scaling.
Table 1: Shown are, for a given target difference to the optimal function value, the number of successful trials (\(n_s\)); the expected running time to surpass \(f_{\text{opt}} + \Delta f\) (IERT, see Figure 1); the 10\%-tile and 90\%-tile of the bootstrap distribution of IERT; the average number of function evaluations in successful trials \(se\), if none was successful, as last entry the median number of function evaluations to reach the best function value \(\text{RT}_{\text{best}}\). If \(f_{\text{opt}} + \Delta f\) was never reached, figures in \textit{italics} denote the best achieved \(\Delta f\)-value of the median trial and the 10\%- and 90\%-tile trial. Furthermore, \(N\) denotes the number of trials, and \(\text{mFE}\) denotes the maximum of number of function evaluations executed in one trial. See Figure 1 for the names of functions.

Figure 2: Empirical cumulative distribution functions (ECDFs), plotting the fraction of trials versus running time (left subplots) or versus \(\Delta f\) (right subplots). The thick red line represents the best achieved results. Left subplots: ECDF of the running time (number of function evaluations), divided by search space dimension \(D\), to fall below \(f_{\text{opt}} + \Delta f\) with \(\Delta f = 10^k\), where \(k\) is the first value in the legend. Right subplots: ECDF of the best achieved \(\Delta f\)-divided by \(10^k\) (upper left lines in continuation of the left subplot), and best achieved \(\Delta f\)-divided by \(10^k\) for running times of \(D,10,D,100,D,1000\), function evaluations (from right to left cycling black-cyan-magenta). Top row: all functions; second row: separable functions; third row: \(\text{mix}\); moderate functions; fourth row: ill-conditioned functions; fifth row: multi-modal functions with adequate structure; last row: multi-modal functions with weak structure. The legends indicate the number of functions that were solved in at least one trial. \(\text{Fevals}\) denotes number of function evaluations, \(D\) and \(\text{DIM}\) denote search space dimension, and \(\Delta f\) and \(\text{DF}\) denote the difference to the optimal function value.
The BBOB (Noiseless) Test Functions

- all define a "scientific question"
  the relevance can hardly be overestimated
- functions are not perfectly symmetric
  and are locally deformed
- 24 functions within five sub-groups
  - separable functions
  - essential unimodal functions
  - ill-conditioned unimodal functions
  - multimodal structured functions
  - multimodal functions with weak or without structure
How do we measure performance?
Measuring Performance

...empirically...

convergence graphs is all we have to start with

Measure here: Run length or runtime or first hitting time (in #funevals) to a given target function value.
ECDF:
Empirical Cumulative Distribution Function of the Runtime

aka

Data Profiles
First Hitting Time is Monotonous
15 Algorithm Runs

![Graph showing 15 Algorithm Runs](image)

- Y-axis: *function value*
- X-axis: \( \log_{10}(\text{function evaluations}) \)
15 Algorithm Runs

\[
\text{function value} \quad \text{log}_{10}(\text{function evaluations})
\]

target
Empirical CDF

- The **ECDF** of run lengths (runtimes) to reach the target
  - has for each data point a **vertical step** of constant size
  - displays for each x-value (budget) the count of observations to the left (first hitting times)

  60% of the runs need between 2000 and 4000 evaluations

  80% of the runs reached the target
Reconstructing Single Runs: Recording More Targets
Reconstructing Single Runs: Recording More Targets

50 equally spaced targets

---

© Dimo Brockhoff, INRIA Lille – Nord Europe  
EMO tutorial, La Rochelle, November 5, 2014  
72
Reconstructing Single Runs: Recording More Targets

![Graph showing function value over log10(function evaluations)]
Reconstructing Single Runs: Recording More

![Graph showing a downward trend in function value with increasing log10 of function evaluations.](image)

**function value**

**log_{10}(function evaluations)**
the empirical CDF makes a step for each star, is monotonous and displays for each budget the fraction of targets achieved within the budget.
ECDF Recovers the Convergence Graph

...discretized and flipped
ECDF Recovers the Convergence Graph

...discretized and flipped
Aggregation of Optimization Runs

function value

\[ \log_{10}(\text{function evaluations}) \]
Aggregation of Optimization Runs

![Graph showing the aggregation of optimization runs](image.png)

- 15 runs
- 50 targets

function value vs. $\log_{10}(\text{function evaluations})$
Aggregation of Optimization Runs

![Graph showing function value vs. log10(function evaluations)]
Aggregate of Optimization Runs

function value

$\log_{10}(\text{function evaluations})$
Exemplary Results
Unimodal vs. Multimodal

20-D unimodal

20-D multimodal

Proportion of problems vs. Running length / dimension

© Dimo Brockhoff, INRIA Lille – Nord Europe
EMO tutorial, La Rochelle, November 5, 2014
Announcement

CEC Special Session on
Black Box Optimization Benchmarking
(CEC-BBOB’2015)

organizers:
Y. Akimoto, A. Auger, D. Brockhoff, N. Hansen,
O. Mersmann, P. Pošík

submission deadline: December 19, 2014

http://coco.gforge.inria.fr/
Back to
Multi-Objective Benchmarking
Data Profiles

exemplary problem:
\[ f_{10}: \text{Ellipsoid} \quad + \quad f_{21}: \text{Gallagher 101 peaks} \]

hypervolume difference
of all non-dominated solutions found to ref. set
More Problems
Aggregation Over All BBOB Problems (300 total)

all functions (F1) & all functions (F2), 5D

F1: $f_1 \rightarrow f_{24}$
F2: $f_1 \rightarrow f_{24}$

Proportion of (function+target) pairs

# f-evaluations / dimension

5-D

all functions (F1) & all functions (F2), 20D

F1: $f_1 \rightarrow f_{24}$
F2: $f_1 \rightarrow f_{24}$

Proportion of (function+target) pairs

# f-evaluations / dimension

20-D
Aggregation Over Function Groups

the only groups where MO-CMA-ES is always outperformed contain separable functions
Conclusions: EMO as Interactive Decision Support

- **modeling**
- **problem**
- **specification** → **optimization** → **adjustment**
- **analysis** → **visualization** → **solution**
- **decision making**
- **preference articulation**
- **optimization**
- **adjustment**
The EMO Community

Links:

- EMO mailing list: https://lists.dei.uc.pt/mailman/listinfo/emo-list
- MCDM mailing list: http://lists.jyu.fi/mailman/listinfo/mcdm-discussion
- EMO bibliography: http://www.lania.mx/~ccoello/EMOO/
- EMO conference series: http://www.dep.uminho.pt/EMO2015/

Books:

- *Multi-Objective Optimization using Evolutionary Algorithms*
  Kalyanmoy Deb, Wiley, 2001
- *Evolutionary Algorithms for Solving Multi Evolutionary Algorithms for Solving Multi-Objective Problems Objective Problems*
  Carlos A. Coello Coello, David A. Van Veldhuizen & Gary B. Lamont, Kluwer, 2nd Ed. 2007
- *Multiobjective Optimization—Interactive and Evolutionary Approaches*
- and more…
PISA: http://www.tik.ee.ethz.ch/pisa/

and many more:
- jmetal
- Shark
- MOEA Framework,
  ...
download of selectors, variators and performance assessment

this page contains the currently available variators and selector (see also principles of PISA) as well as performance assessment tools (see also performance assessment). The variators are mainly test and benchmark problems that can be used to assess the performance of different optimizers. EXPO is a complex application form the area of computer design that can be used as a benchmark tool too. The selectors are state-of-the-art evolutionary multi-objective optimization methods. If you want to write or submit a module, please look at write and submit a module. Links to documentation on the PISA specification can be found at documentation.

Jarojlov Hajek pointed out a severe bug in the WFG selector, please redownload the module if your version is older than 2010/02/03.

Optimization Problems (variator)

GWLAB - Multi-Objective Groundwater Management

Package: in Matlab

LOTZ - Demonstration Program

Source: in C
Binaries: Solans, Windows, Linux

LOTZ2 - Leading Ones Trailing Zeros

Source: in C
Binaries: Solans, Windows, Linux

LOTZ2 - Java Example Variator

Source: in Java
Binaries: Windows, Linux

Knapsack Problem

Source: in C
Binaries: Solans, Windows, Linux

EXPO - Network Processor Design Problem

Optimization Algorithms (selector)

SPAM - Set Preference Algorithm for Multiobjective Optimization

Source: in C
Binaries: Windows, Linux 32bit, Linux 64bit

SHV - Sampling-based HyperVolume-oriented algorithm

Source: in C
Binaries: Windows, Linux 32bit, Linux 64bit

SIBEA - Simple Indicator Based Evolutionary Algorithm

PISA: http://www.tik.ee.ethz.ch/pisa/
References
References


References


