Technology
Arts Sciences
TH Köln

Benchmarking and Experimentation:
Pitfalls and Best Practices

CEC 2021

Mike Preuss, Boris Naujoks, Thomas Bartz-Beielstein

Who are we?
Prof. Dr. Thomas Bartz-Beielstein

- AI expert with 30+ years of experience
- Professor for applied mathematics at TH Köln
- Director Institute for Data Science, Engineering, and Analytics (IDE+A)
- Research: AI, ML, simulation, optimization
- Founder of the Sequential Parameter Optimization (SPO), https://www.spotseven.de
- Applications: water industry, elevator control, automotive, mechanical engineering, ...

Who are we?
Boris Naujoks

- Professor for applied mathematics at TH Köln
- Vice Director Institute for Data Science, Engineering, and Analytics (IDE+A)
- Co-Founder of Institute for Innovative Pharmaceuticals for the Aging Society (InnovAGe)
- Research: AI, EMO/MCDM, surrogate assisted optimization, benchmarking
- Applications: engineering design, drug design, games

Who are we?
Mike Preuss

- Assistant Prof Universiteit Leiden (NL)
- Doing lots of things with AI:
  - Evolutionary Optimization
  - Game AI (from Games to real world and vice versa)
  - Social Media Computing
- Talks about experimental methodology, benchmarking, parameter setting since 2003
- Possibly well-known for Nature paper on chemical retrosynthesis (transfers AlphaGo)
- PhD TU Dortmund (DE) 2013 on Evolutionary Multimodal Optimization

Shonan Seminar on Game AI, Japan 2019

Who are you?

- Starting to work experimentally, looking for orientation?
- Or more experienced, looking for updates?

- We try to give you an overview and links to follow
- But the topic is too big to learn it from scratch in 1h
- And it cannot be learned theoretically (by reading papers, very much like climbing)
- It is important to experience it on your own

Image by Laurence Derippe from Pixabay

Overview

- Introduction to Experimentation
- Introduction to Benchmarking
- Multi-objective Setting
- Single-objective Setting
- Hyperparameter Tuning Setting
- Performance Measurement
- Results Analysis
- Results Presentation
- Reproducibility
- Pitfalls
- Summary

What is an experiment?

Wikipedia (en):

An experiment is a method of testing - with the goal of explaining - the nature of reality. [...]
More formally, an experiment is a methodical procedure carried out with the goal of verifying, falsifying, or establishing the accuracy of a hypothesis.

keywords: goal, reality, methodical procedure, hypothesis

The theory "wars"

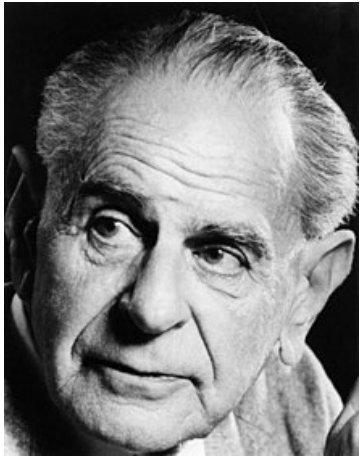
- up to today, I see reviews aiming at rejecting papers because they are “purely empirical and lacking theory”
- the term “empirical” is not wrong, but disregards that we have control, we do it “experimentally”
- in many areas in computer science, research is Either theoretical Or experimental
- but ideally, it shall be both interacting with each other
- do you think experiments are easy? this is from the foreword of the book on the left:

However, experiments require a lot of work, so the reader may be warned: Performing a good experiment is as demanding as proving a new theorem.
Dortmund, November 2005, Hans-Paul Schwefel

Picture from Momentmal on Pixabay

Science based on empiricism

- Popper rejected the classical inductivist view in favor of the *empirical falsification*
- Theories cannot be proven, but they can be falsified: experiments shall attempt to contradict a theory
- If something cannot be falsified in principle, it is not a scientific theory
- Modern statistics (statistical testing) goes along with this: reasoning is indirect, you falsify hypotheses
- He rejects also logical justification of induction: just because something has always happened, it is not guaranteed to happen again



Karl Popper
Picture from Wikimedia Commons

It is not so easy: Rosenthal effect

- Rosenthal/Fode: expectations of experimenters can lead to wrong conclusions
- they gave rats from the same origin to two groups of students to test them
- students were told that “their” rats were especially intelligent or stupid
- this was actually reported by students as conclusions of experiments albeit *not true*
- Rosenthal/Jacobson showed similar for “primed” primary school teachers when testing IQ of pupils
- there are more effects like this, advice from me: “never watch a running experiment”

Picture by sipa from Pixabay

The success bias

- what you see is only what works (at least has been working at least once for one problem)
- largely no negative results are published
- the process to obtain one positive result can be long and tedious
- example: to arrive at AlphaGo has required years of research, failure, and lots of intermediate steps
- replicating successful results is often not possible due to under-specification

picture from luvmybry on Pixabay

Ingredients for a good experiment

- fairness (even if we want to show that our method is better)
- openness (provide the means to get surprised)
- defined targets
- how do we determine which method is the best (comparison)
- what are the minimal conditions that must be reached?
- defined methodology (not ad-hoc)
- documentation (sufficient for replication)
- iteration (the first research question/hypothesis is usually not very good)

Marie Curie

What is Benchmarking?

Wikipedia (en):

Benchmarking is the practice of comparing business processes ... Dimensions typically measured are quality, time and cost.

...

Benchmarking is used to measure performance using a specific indicator (...) resulting in a metric of performance that is then compared to others.

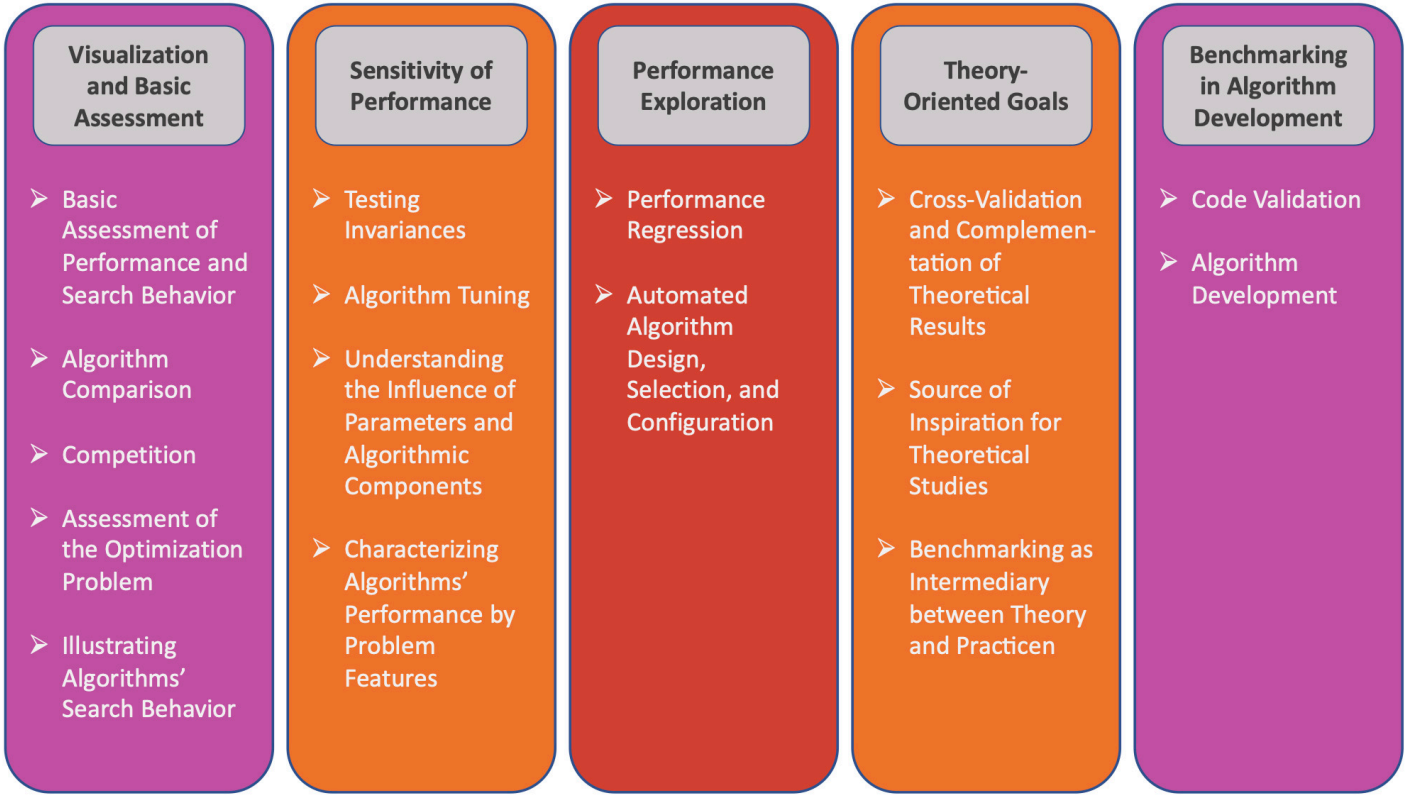
Keywords: Comparing, measure performance, specific indicator

Benchmarking: what for?

- Compare performance (of algorithms, in particular metaheuristics)
  - Given different parameterizations
  - With other algorithms
  - For special problems
- Caution:

  We cannot do as many tests as we should on real problems with new algorithms

Common Goals of Benchmarking Studies



From real-world to benchmark and back

- With benchmarking we want to prepare for real-world situations

Choices:

- What are the objectives?
- Choose "the right" algorithm
- What are suitable parameters?

Outlook:
Three Settings – Three Perspectives

- Different needs, different states-of-the-art, different practices

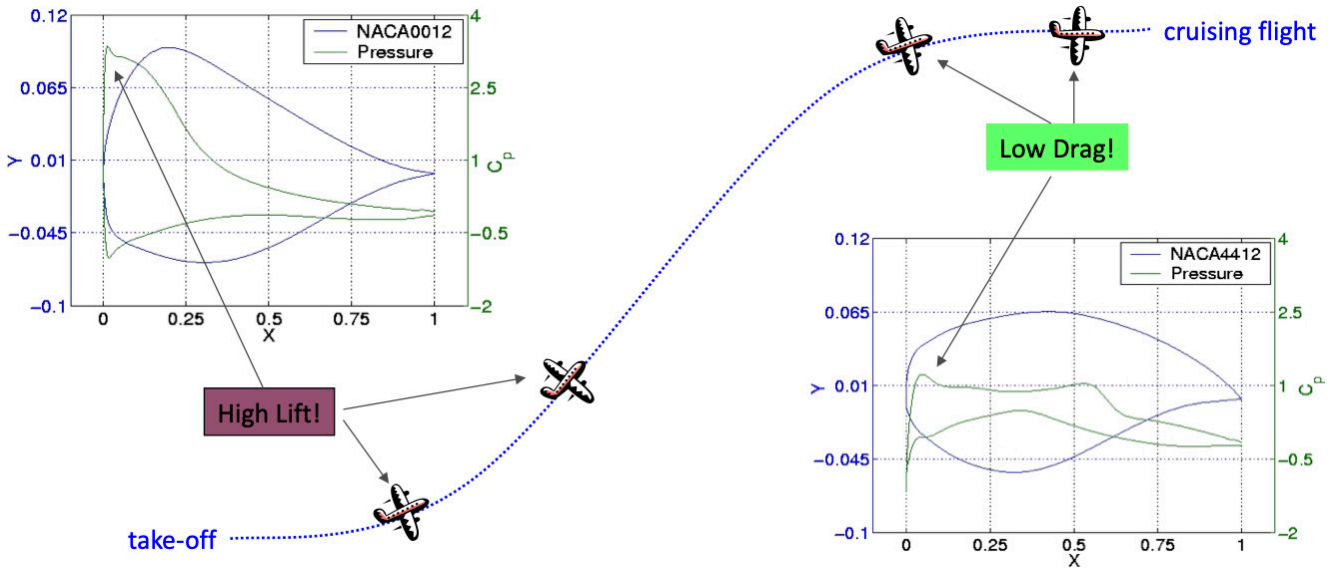
- Multi objective optimisation (Boris): Airfoil design, measure performance
- Single objective optimisation (Mike): from Linearjet to Nevergrad/Shiwa
- Hyperparameter tuning (Thomas): SPOT – Deep Learning

Multi objective optimization: Why?

- Aggregation of objectives still standard
- MCDM techniques from Operations Research exist since 50+ years
  - Common techniques: aggregation e.g. using weighted sum approach (proven to fail in some situations)
- Benefits from MCDM / EMO
  - Gain problem understanding
  - Discover interdependencies and trade-offs of objectives
  - Avoid uncertainties

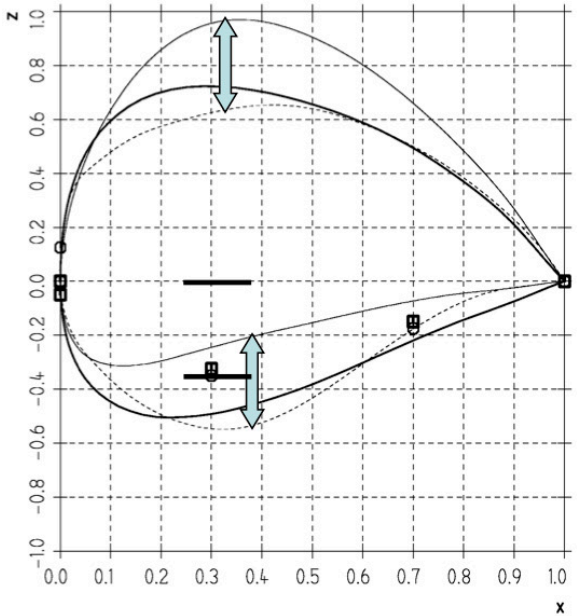
Multi-objective Airfoil Design

- Design of an optimal airfoil for different flight conditions



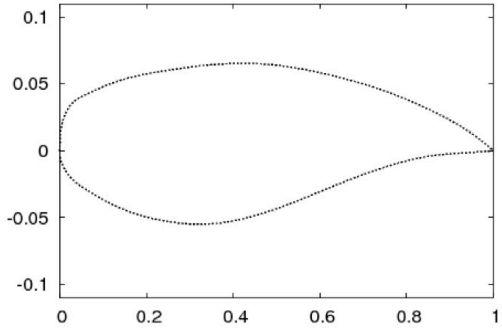
Multi-objective Airfoil Design - Preconditions

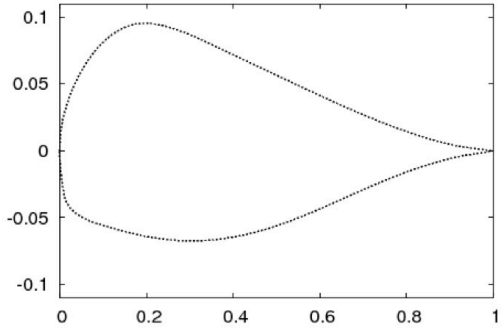
- Provided by industrial partner
  - Airfoil parameterisation
    - 18 parameters, i.e. coordinates of Bezier splines
    - Box constraints for all parameters
  - Simulation environment (CFD based, Navier-Stokes, several minutes)
  - Post-processing providing 2 objectives (under fixed flight conditions for given airfoil)
    - Lift coefficient (to maximise)
    - Drag coefficient (to minimise)
  - 1000 evaluations allowed per optimisation run! (approximately)



Multi-objective Airfoil Design - How to set up optimisation

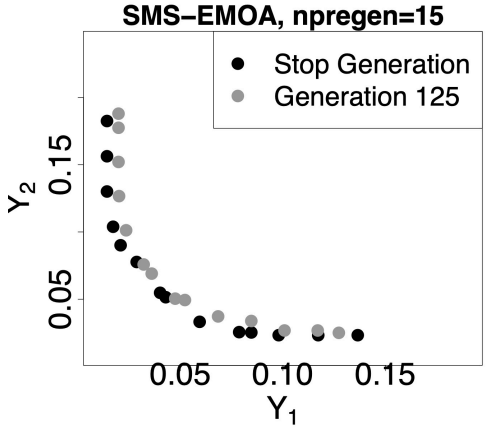
- 2 – objective problem: NSGA-II (SMS-EMOA)
  - Not the best, but most popular
  - Easy to use (and understand, relates to "Law of the Instrument")
- Parameterization?
  - According to preconditions (all not optimal
  - $(\mu + \mu)$ - approach, μ = 10 (parameter tuning ...)
  - Stopping criterion: 100 generations (performance dependent ...)
  - Standard variation operators: SBX + PM (but how to choose parameters? Use standard settings, parameter tuning again ...)

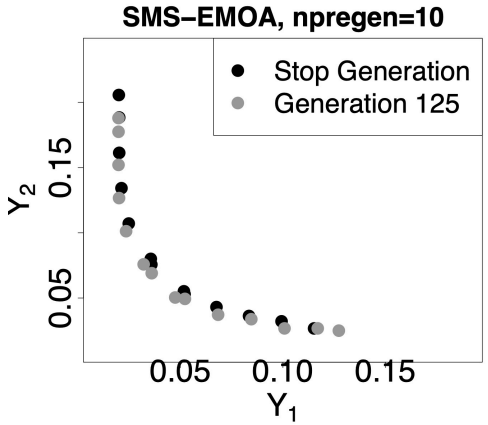




Multi-objective Airfoil Design - How to analyze results

- 2 – objective problem
  - Visual inspection possible but not always easy
  - Multiple runs, figures become more complex
  - Techniques like PF aggregation didn't make their way
- 3 – objective problem
  - Even harder, rotations needed
- Utility function needed
  - Hypervolume, IGD (de-facto standards)

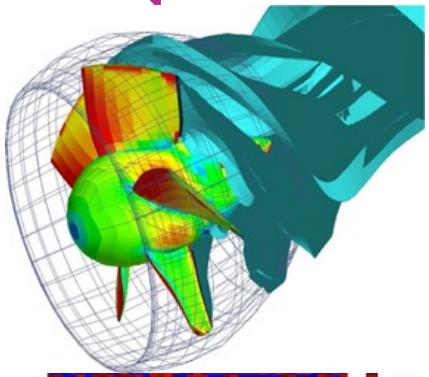


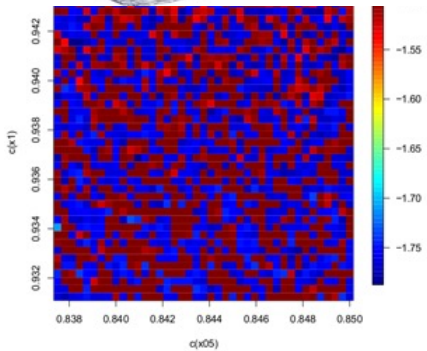


Multi-objective Airfoil Design - Questions

- How about more than 2 objectives?
  - Many objective optimization (MaOO)
- Considering Surrogates
  - How to integrate?
  - How to handle errors?
- How about other aspects, structure etc.
  - Multi disciplinary design (MDO)

Linearjet: single-objective expensive simulation-based industrial optimization

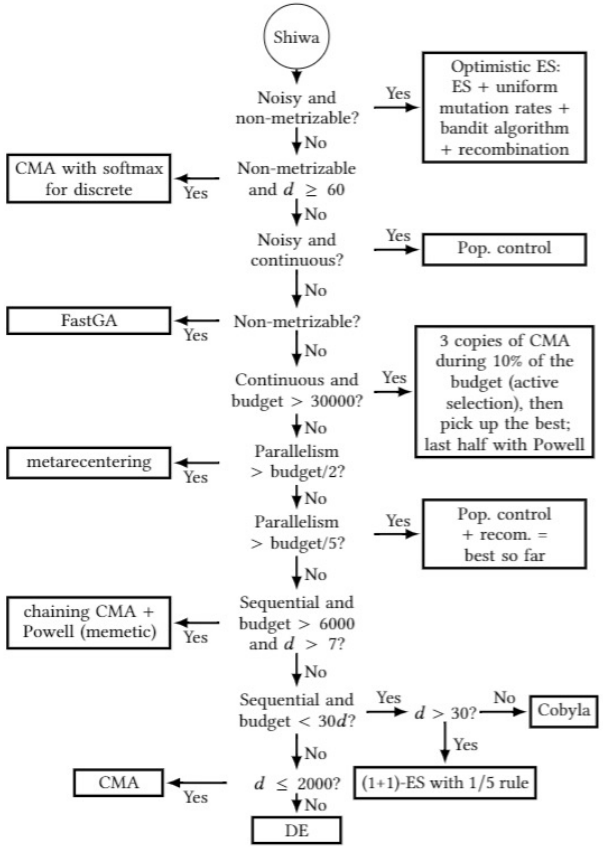




- Concrete problem here: minimize cavitation (underpressure bulbs)
- Typical industrial optimization setting: around 20 design parameters
- Complex setup of 26 simulation tools chained, took 2 years to set up
- Runtimes for high accuracy: hours, low accuracy: minutes
- Landscape 2D cuts partly flat, partly chaotic
- Benchmarking angle: from low to high accuracy, algorithm calibration

Nevergrad / Shiwa

- Shiwa is a toolbox-based multi-algorithm using best matching parts of Nevergrad
- Benchmarking has played a major role in putting it together
- The exact conditions and their sequence are manually calibrated -> tuning is possible
- Can be compared and improved with benchmarking
- Probably too complex to interpret parameter interactions: made for real-world application



Jialin Liu, Antoine Moreau, Mike Preuss, Jérémy Rapin, Baptiste Rozière, Fabien Teytaud, Olivier Teytaud: Versatile black-box optimization. GECCO 2020

Hyperparameter Tuning with SPOT

- Surrogate-model based hyperparameter tuning
- Deep-learning models require the specification of several architecture-level parameters: hyperparameters
- Hyperparameters to be distinguished from the parameters of a model that are optimized during the training phase (backprop)
    - Which dropout rate should be used?
    - How many layers should be stacked?
    - How many filters (units) should be used in each layer?
    - Which activation function should be used?

Goals

- Understanding: in contrast to standard HPO approaches, SPOT provides statistical tools for understanding hyperparameter importance
- Transparency and Explainability: understanding is a key tool for enabling transparency, e.g., quantifying the contribution of deep learning components (layers, activation functions, etc.).
- Reproducibility

Tuning versus Benchmark Studies

- Tuning: one algorithm (neural network) and one problem
  - Improve hyperparameters
  - Understand one algorithm in one specific setting
  - Compare to baseline (random search, random sampling,..)
- Benchmark: several algorithms on one or several problems

Softwaretools

- R – statistical programming language
- Tensorflow
- Keras
- SPOT

Typical questions regarding hyperparameters in DL

- How many layers should be stacked?
- Which dropout rate should be used?
- How many filters (units) should be used in each layer?
- Which activation function should be used?

If you want to get to the very limit of what can be achieved on a given task, you can't be content with arbitrary [hyperparameter] choices made by a fallible human. Your initial decisions are almost always suboptimal, even if you have good intuition. You can refine your choices by tweaking them by hand and retraining the model repeatedly—that's what machine-learning engineers and researchers spend most of their time doing. But it shouldn't be your job as a human to fiddle with hyperparameters all day—that is better left to a machine.

HPT

- HPT develops tools to explore the space of possible hyperparameter configurations *systematically*, in a structured way.
- Essential for this process is the HPT algorithm that uses the history of validation performance, given various sets of hyperparameters, to choose the next set of hyperparameters to evaluate
- Updating hyperparameters is extremely challenging, because computing the hyperparameter response surface can be very expensive

HPT Approaches

- manual search
- random search
- grid and pattern search
- model free algorithms, i.e., algorithms that do not explicitly make use of a model, e.g., EAs
- hyperband (multi-armed bandit strategy)
- Surrogate Model Based Optimization (SMBO) such as Sequential Parameter Optimization Toolbox

HPT Goals

- (R-1) Goals: what are the reasons for performing HPT?
- (R-2) How to select suitable problems? Surrogates?
- (R-3) Algorithms: how to select a portfolio of DL algorithms ?
- (R-4) Performance: how to measure performance?
- (R-5) Analysis: how to evaluate results?
- (R-6) Design: how to set up a study?
- (R-7) Presentation: how to describe results?
- (R-8) Reproducibility?

HPT Optimizers

- If two optimizers have an inclusion relationship, the more general optimizer can never be worse with respect to any metric of interest, provided the hyperparameters are sufficiently tuned to optimize that metric [Choi et al. 2019].
  - SGD ⊆ Momentum ⊆ RMSProp
  - SGD ⊆ Momentum ⊆ Adam
  - SGD ⊆ Nesterov ⊆ NAdam
- For example, MOMENTUM can be approximated with ADAM.

HPT Performance

- Metrics
  - training loss
  - training accuracy
  - validation loss
  - validation accuracy
  - test loss
  - test accuracy
- $\psi_i(train) < \psi_j(train) \not\Rightarrow \psi_i(test) < \psi_j(test)$

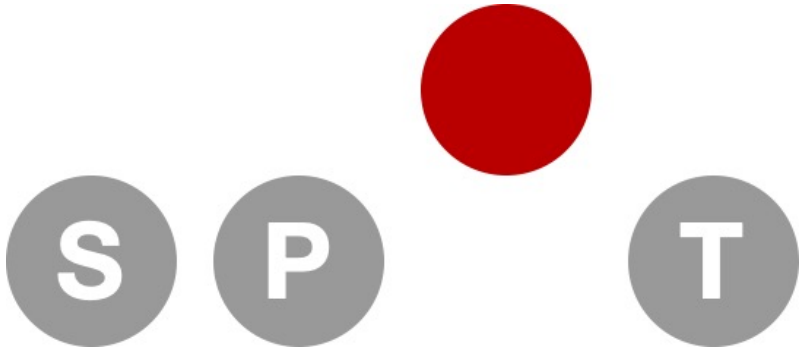
HTP Performance

- Estimation of test error for a particular training set is not easy in general, given just the data from that same training set.
- Instead, cross-validation and related methods may provide reasonable estimates of the expected error.
- For a *relative comparison* between models during the tuning procedure, in-sample error is convenient and often leads to effective model selection.
- The reason is that the relative (rather than absolute performance) error is required for the comparisons.

HPT Performance

- *Result from* Choi et al. [2019] :
  - *final results hold regardless of whether they compare final validation error, i.e., ψ(val), or test error, i.e., ψ(test)*

DEMO



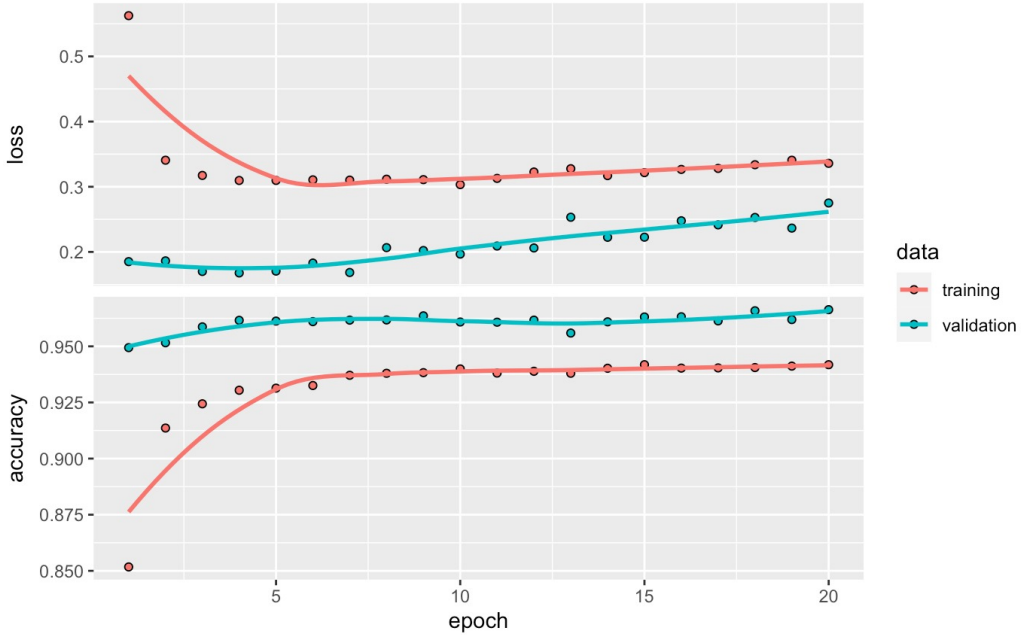
https://cran.r-project.org/package=SPOT
https://www.spotseven.de

Bartz-Beielstein,T. Surrogate model based hyperparameter tuning for deep learning with SPOT. Preprint. 2021. Available via https://www.spotseven.de/new-publications/
(and later this week on arXiv)

SPOT

- Initially, a population of (random) solutions is created.
- A set of surrogate models is specified.
- Then, the solutions are evaluated on the objective function.
- Next, surrogate models are built.
- A global search is performed on the surrogate model(s) to generate new candidate solutions.
- The new solutions are evaluated on the objective function, e.g., the loss is determined.

SPOT Metrics



SPOT Hyperparameters

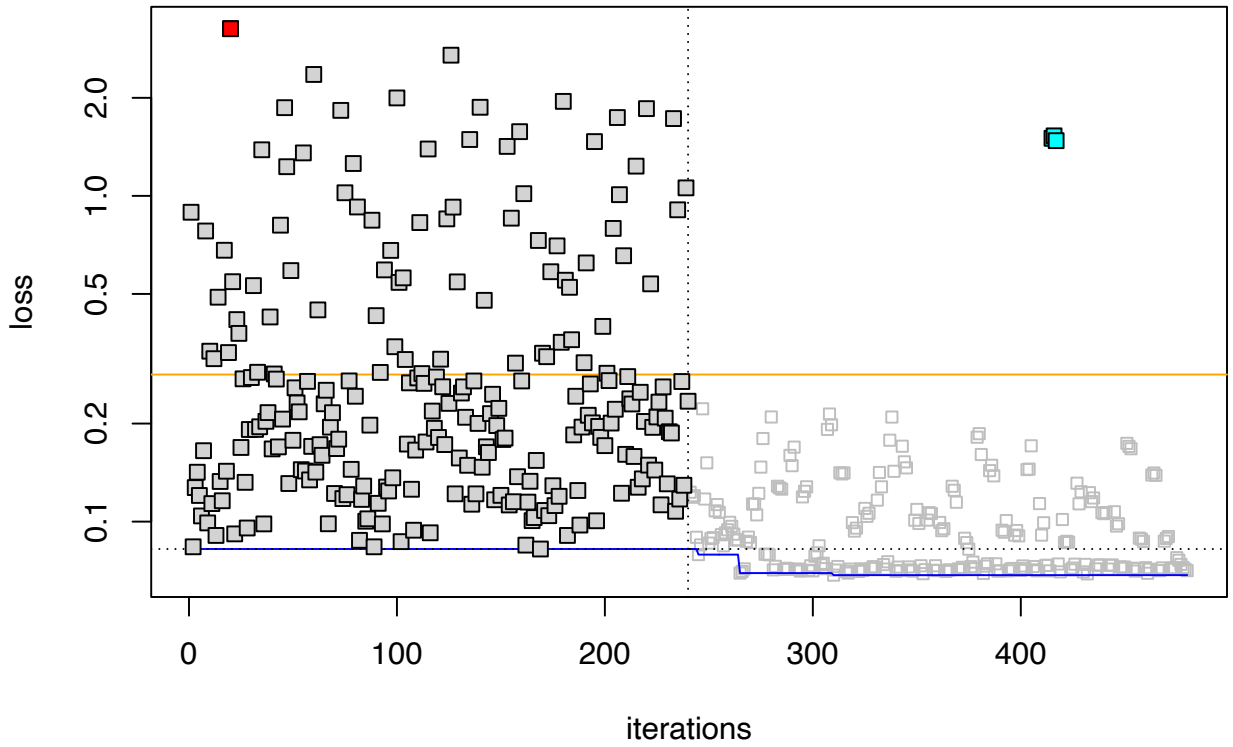
Table 3: The hyperparameters and architecture choices for the first DNN example: fully connected networks

Variable Name	Hyperparameter	Type	Default	Lower Bound	Upper Bound
x_1	first layer dropout rate	numeric	0.4	$1e-6$	1
x_2	second layer dropout rate	numeric	0.3	$1e-6$	1
x_3	units per first layer	integer	256	16	512
x_4	units per second layer	integer	128	4	256
x_5	learning rate	numeric	0.001	0.0001	0.1
x_6	training epochs	integer	20	5	25
x_7	batch size	integer	64	8	256
x_8	rho	numeric	0.9	0.5	0.999

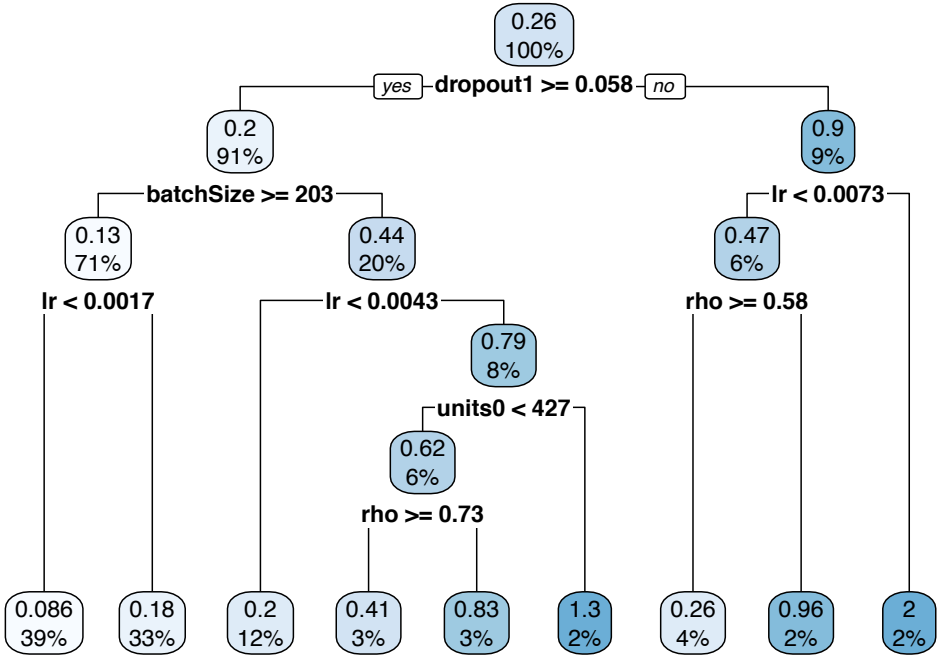
SPOT Interface

```
res <- spot(
    x = NULL,
    fun = funTfruns,
  lower <- c(1e-6, 1e-6, 16, 16, 1e-9, 10, 16, 0.5),
  upper <- c(0.5, 0.5, 512, 256, 1e-2, 50, 512, 1-1e-3),
    control = list(
        funEvals = 480,
        types = c(
            "numeric",
            "numeric",
            "integer",
            "integer",
            "numeric",
            "integer",
            "integer",
            "numeric"
        )
    )
)
```

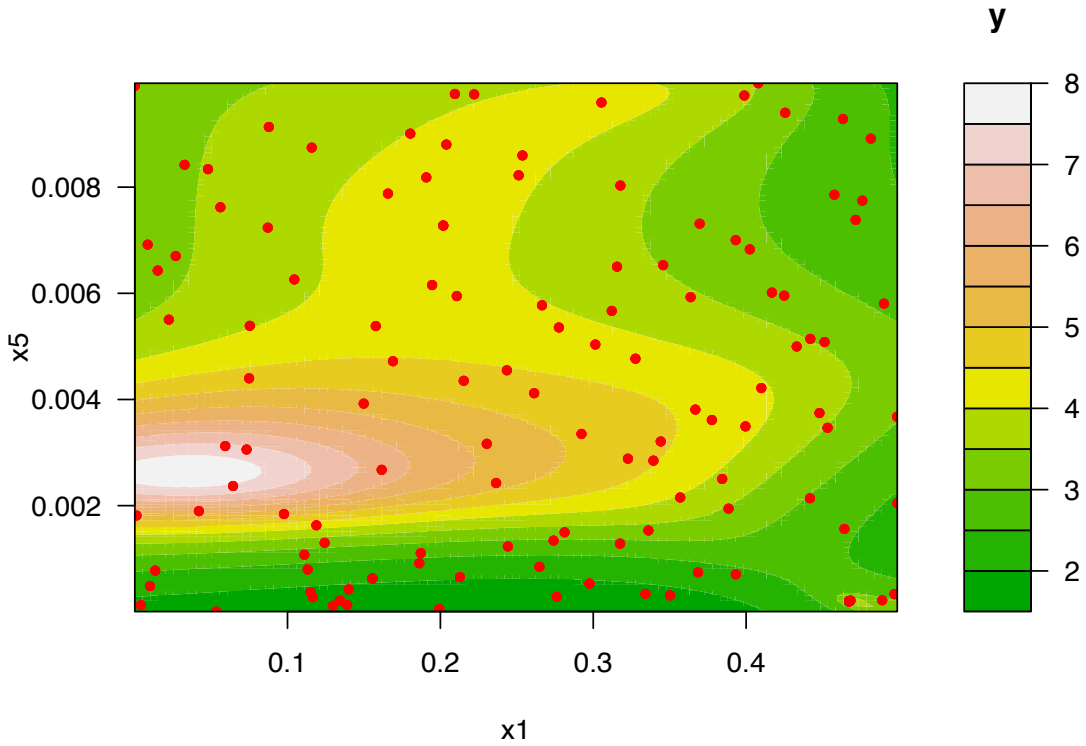
SPOT Results



SPOT Results



SPOT Results

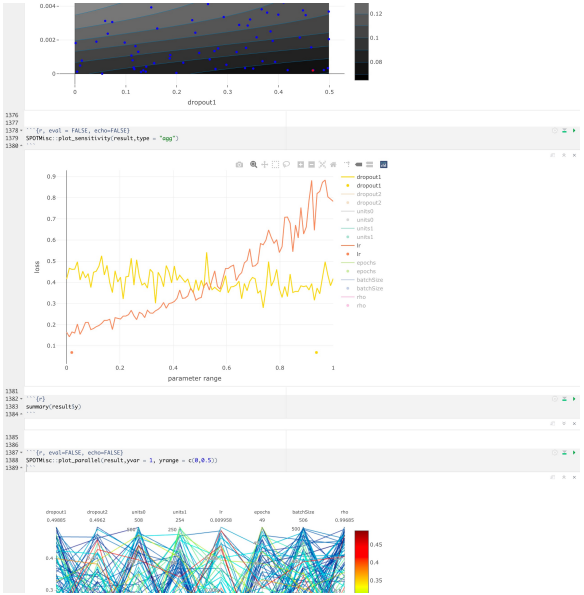


SPOT: Datascope

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.687e-01  1.005e-01   6.653 7.97e-11 ***
res$x1      -4.783e-01  9.330e-02  -5.126 4.32e-07 ***
res$x2       2.856e-01  1.007e-01   2.837  0.00475 **
res$x3       4.506e-04  1.034e-04   4.357 1.62e-05 ***
res$x4      -1.798e-04  2.181e-04  -0.824  0.41010
res$x5       4.716e+01  4.353e+00  10.832  < 2e-16 ***
res$x6       7.589e-03  1.224e-03   6.201 1.23e-09 ***
res$x7      -8.828e-04  9.885e-05  -8.931  < 2e-16 ***
res$x8      -6.831e-01  9.154e-02  -7.462 4.14e-13 ***
---
```

SPOT: Interactive Graphics

- Live demo
- RStudio

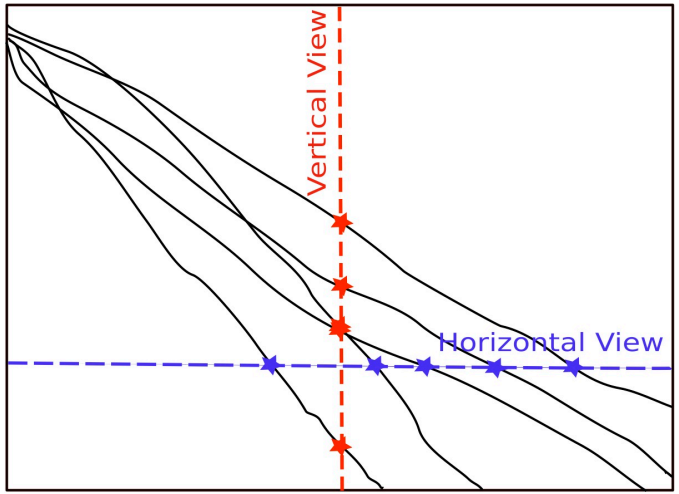


SPOT: Observations as hypotheses

- HPT can be used as a datascope for the optimization of DNN hyperparameters.
- Observations can be stated as hypotheses, e.g.,
- (H-1) hyperparameter x1, i.e., the dropout rate, has a significant effect on the loss function. Its values should larger than zero.
- More:
    - Bartz- Beielstein,T. Surrogate model based hyperparameter tuning for deep learning with SPOT. Preprint. 2021. Available via https://www.spotseven.de/new-publications/ (and later this week on arXiv)

Performance measurement: Single Objective

- Traditionally most of performance measuring in vertical view: not comparable!
- Horizontal view provides comparability: algorithm x needs y times as long to target
- For meaningful analysis we need multiple targets, e.g. 50 as in COCO/BBOB
- Multiple runs (at least 20 for stats tests), ideally on slightly different instances



Picture from COCO manual
https://coco.gforge.inria.fr/COCOdoc/bbo_experiment.html

Expected Runtime / Success Performance

- How to measure performance if runs fail?
- Expected Runtime (ERT) uses average evaluatios to target and success chance

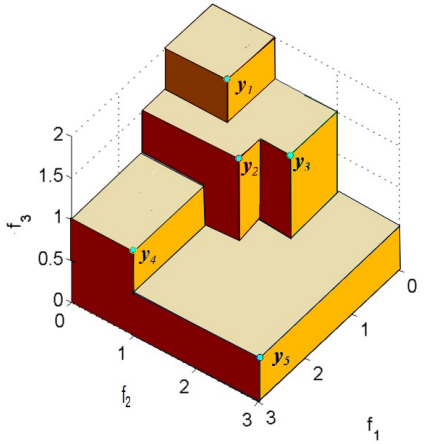
- If you want to directly compare 2 algorithms and have no failures, use Wilcoxon's rank sum test or similar
- Nonparametric tests make less assumptions
- Do not blow up number of repetitions to get a good p-value, fix this in advance

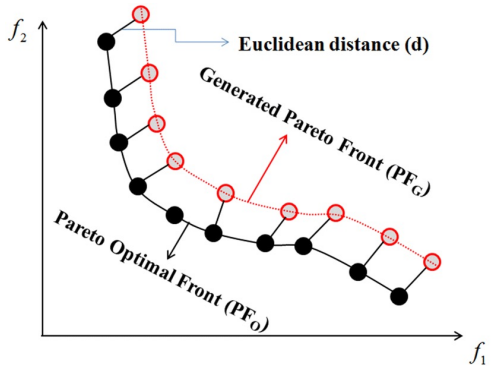
$$\begin{aligned} \mathrm{ERT}(f_{\mathrm{target}}) &= \mathrm{RT_S} + \frac{1-p_\mathrm{s}}{p_\mathrm{s}}\mathrm{RT_{US}} \\ &= \frac{p_\mathrm{s}\mathrm{RT_S} + (1-p_\mathrm{s})\mathrm{RT_{US}}}{p_\mathrm{s}} \\ &= \frac{\#\mathrm{FEs}(f_{\mathrm{best}} \geq f_{\mathrm{target}})}{\#\mathrm{succ}} \end{aligned}$$

from COCO manual
https://coco.gforge.inria.fr/COCOdoc/bbo_experiment.html

Performance measurement: Multi Objective

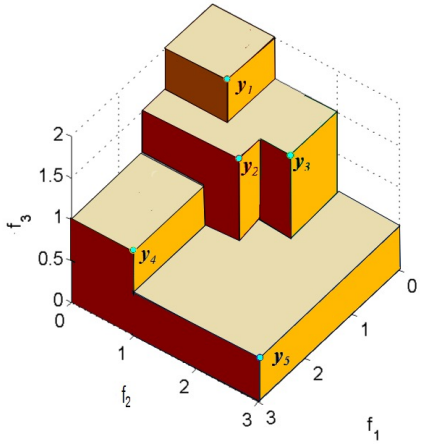
- Utility functions commonly used
- Hypervolume
  - Pareto compliant but computationally expensive
- IGD – Inverted Generational Distance (+)
  - Reference set of (near) optimal solutions needed

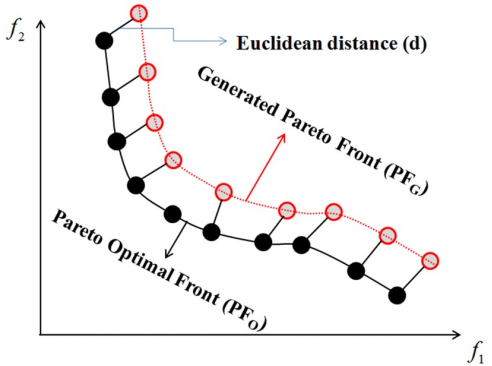




Multi Objective algorithm comparison

- Even worth
- No agreed (good) suite of test functions
  - Mainly rather old ones
- Limited number of real world test functions
  - No standard suites
- No standards, good references, data sets etc.
- Even more, deeper pitfalls …





Results analysis

- Different ways to analyze results ?!? Sure!
- However, multi-objective problems can be mapped to single-objective ones using utility functions
- Does it pay of?
  - Not sure, tools for single-objective rarely used to evaluate multi-objective problems
  - MOO (like 2) decades behind SO optimization? MaOO even more!

Considerations: Problem Design

- Sound test problem specification
  - train, validation, and test set
  - initialization (starting points in optimization)
  - Surrogate benchmarks (Deep Network + CFD)
- Repeats (power of the test, severity)
- Meaningful difference (e.g. Pareto front comparison)
- Scientific relevance != statistical significance
- Avoid floor and ceiling effects:
  - Compare to baseline (random search, random sampling, mean value...)

Considerations: Algorithm Design

- Sound algorithm (neural network) specification
  - Initialization, pre-training (starting points in optimization)
  - Hyperparameter (ranges, types)
  - Additional (untunable) parameters
  - Noise
- Reproducibility
  - Last but not least: open source

Reporting and keeping track of experiments

around 40 years of experimental tradition in Computational Intelligence/ML, but:

- no standard scheme for reporting experiments (experimental protocols)
- instead: one (“Experiments”) or two (“Experimental Setup” and “Results”) sections in papers, often providing a bunch of largely unordered information
- affects readability and impairs reproducibility

keeping experimental journals helps:

- record context and rough idea
- report each experiment
- running where (machine)
- finished when (date/time), link to result file(s)

⇒ we suggest a 7-part reporting scheme (actually sort of borrowed from Physics)

Experimental report

suggested structure:

1. **research question**: what do we investigate?
2. **pre-experimental planning** – first explorative ad-hoc expereriments to find target and setup (parameters etc.)
3. **task** – scientific and related statistical hypotheses – under which conditions is a method “successful”? Important to define prior to experiment!
4. **setup** – exact setup of an experiment that enables replication
5. **results/visualizations** – tables, pictures – not interpreted
6. **observations** – peculiarities we find in the results
7. **discussion** – statistical test results, subjective interpretation of results and observations

Reproducibility

- A problem in general (science!)
- Not solved yet
- More complex software: less knowledge on what actually happens
- Needs a lot of available information (code, data, parameters)
- Difficult for proprietary code/data

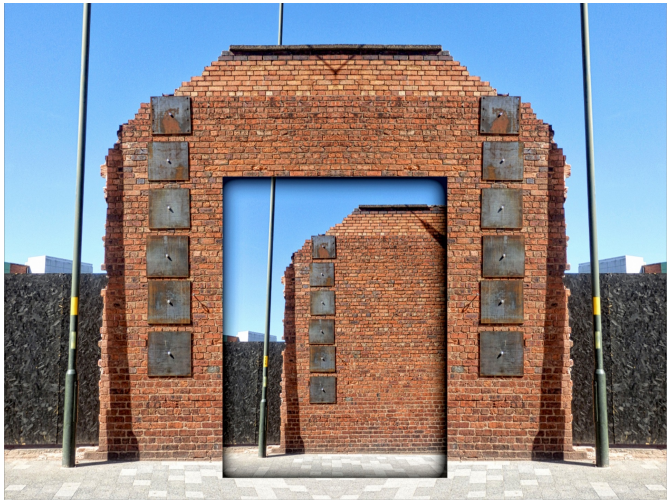


Image by Steve Watts from Pixabay

Replication

- repeatability: same experimenter, same conditions
- reproducibility: different experimenter, same conditions
- these two occur in literature also with opposite meanings
- triangulation: multiple approaches to the same problem
- criticism: most studies are neither repeated nor reproduced

Matters arising

Transparency and reproducibility in artificial intelligence

https://doi.org/10.1038/s41586-020-2766-y
Received: 1 February 2020
Accepted: 10 August 2020
Published online: 14 October 2020
Check for updates

Benjamin Haibe-Kains[1,2,3,4,5], George Alexandru Adam[3,5], Ahmed Hosny[6,7], Farnoosh Khodakarami[1,2], Massive Analysis Quality Control (MAQC) Society Board of Directors*, Levi Waldron[8], Bo Wang[2,3,5,9,10], Chris McIntosh[2,5,9], Anna Goldenberg[3,5,11,12], Anshul Kundaje[13,14], Casey S. Greene[15,16], Tamara Broderick[17], Michael M. Hoffman[1,2,3,5], Jeffrey T. Leek[18], Keegan Korthauer[19,20], Wolfgang Huber[21], Alvis Brazma[22], Joelle Pineau[23,24], Robert Tibshirani[25,26], Trevor Hastie[25,26], John P. A. Ioannidis[25,26,27,28,29], John Quackenbush[30,31,32] & Hugo J. W. L. Aerts[6,7,33,34]

ARISING FROM S. M. McKinney et al. *Nature* https://doi.org/10.1038/s41586-019-1799-6 (2020)

Table 1 | Essential hyperparameters for reproducing the study for each of the three models

	Lesion	Breast	Case
Learning rate	Missing	0.0001	Missing
Learning rate schedule	Missing	Stated	Missing
Optimizer	Stochastic gradient descent with momentum	Adam	Missing
Momentum	Missing	Not applicable	Not applicable
Batch size	4	Unclear	2
Epochs	Missing	120,000	Missing

from *Nature,* 2020

https://arxiv.org/abs/2102.03380

Some guidelines for EC

Reproducibility in Evolutionary Computation

MANUEL LÓPEZ-IBÁÑEZ, University of Málaga, Spain
JUERGEN BRANKE, University of Warwick, UK
LUÍS PAQUETE, University of Coimbra, CISUC, Department of Informatics Engineering, Portugal

Experimental studies are prevalent in Evolutionary Computation (EC), and concerns about the reproducibility and replicability of such studies have increased in recent times, reflecting similar concerns in other scientific fields. In this article, we suggest a classification of different types of reproducibility that refines the badge system of the Association of Computing Machinery (ACM) adopted by TELO. We discuss, within the context of EC, the different types of reproducibility as well as the concepts of *artifact* and *measurement*, which are crucial for claiming reproducibility. We identify cultural and technical obstacles to reproducibility in the EC field. Finally, we provide guidelines and suggest tools that may help to overcome some of these reproducibility obstacles.

Additional Key Words and Phrases: Evolutionary Computation, Reproducibility, Empirical study, Benchmarking

1 INTRODUCTION

As in many other fields of Computer Science, most of the published research in Evolutionary Computation (EC) relies on experiments to justify their conclusions. The ability of reaching similar conclusions by repeating an experiment performed by other researchers is the only way a research community can reach a consensus on a given hypothesis. From an engineering perspective, the assumption that experimental findings hold under similar conditions is essential for making sound decisions and predicting their outcomes when tackling a real-world problem.

The "reproducibility crisis" refers to the realisation that many experimental findings described in peer-reviewed scientific publications cannot be reproduced, either because they lack enough details to repeat the experiment or because repeating the experiment leads to different conclusions. Despite its strong mathematical basis, Computer Science (CS) also shows signs of suffering such a crisis [Cockburn et al. 2020; Fonseca Cacho and Taghva 2020; Gundersen et al. 2018]. EC is by no means an exception. In fact, as we will discuss later, particular challenges of reproducibility in EC arise from the stochastic nature of the algorithms.

Also going to appear in the **ACM Transactions** on Evolutionary **Learning** and **O ptimization** (TELO)

Get involved!

Special Issue on

"Reproducibility in Evolutionary Computation"

Evolutionary Computation Journal, MIT Press

https://direct.mit.edu/evco/pages/submission-guidelines

DEADLINE: November 30, 2021

Guest Editors:

Mike Preuss, Universiteit Leiden, The Netherlands, m.preuss@liacs.leidenuniv.nl

Luís Paquete, University of Coimbra, Portugal, paquete@dei.uc.pt

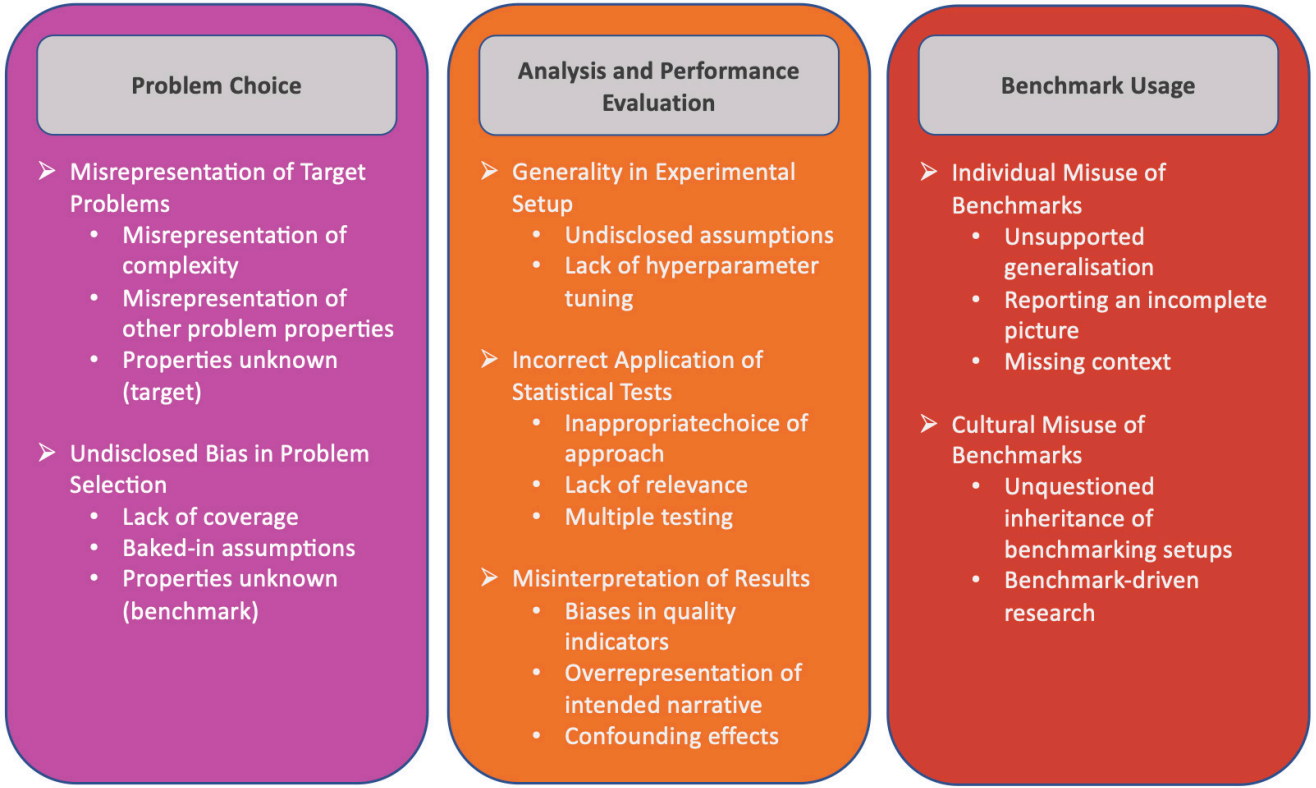
Associate Editor:

Manuel López-Ibáñez, University of Málaga, Spain, manuel.lopez-ibanez@uma.es

Description:

Experimental research is crucial in Evolutionary Computation. The scientific method requires that empirical results are reproducible by the authors themselves and replicable by others. Computer Science in general, and Evolutionary Computation in particular, show signs of a reproducibility crisis despite their digital underpinnings. Interest in improving reproducibility in Computer Science and other empirical sciences has grown in recent years and there is a growing number of works analysing current and best practices, obstacles and guidelines, effectiveness of journal policies, etc. Reproducibility issues in the context of Evolutionary Computation have been a topic of discussion for a long time in the context of best practices for empirical research, but there are few studies analysing reproducibility in EC research and reproducibility studies themselves are extremely rare. There is room for improvement to attain the minimum standards for reproducibility encouraged in other scientific fields. Challenges for reproducibility in EC research arise from the stochastic nature of the

Pitfalls


Problem Choice
Misrepresentation of Target Problems
Misrepresentation of complexity
Misrepresentation of other problem properties
Properties unknown (target)
Undisclosed Bias in Problem Selection
Lack of coverage
Baked-in assumptions
Properties unknown (benchmark)
Analysis and Performance Evaluation
Generality in Experimental Setup
Undisclosed assumptions
Lack of hyperparameter tuning
Incorrect Application of Statistical Tests
Inappropriatechoice of approach
Lack of relevance
Multiple testing
Misinterpretation of Results
Biases in quality indicators
Overrepresentation of intended narrative
Confounding effects
Benchmark Usage
Individual Misuse of Benchmarks
Unsupported generalisation
Reporting an incomplete picture
Missing context
Cultural Misuse of Benchmarks
Unquestioned inheritance of benchmarking setups
Benchmark-driven research

Checklist to avoid Pitfalls

Not fully applicable in all cases
Maybe helpful and practicable guideline for researchers and practitioners

- Start with a hypothesis
- Compute baselines (state-of-the-art, different type of algorithm, random search)
- Learn about target and test problems (visualisation, ELA, evolutionary path)
- Consciously choose test problems that reflect target in appropriate distribution

(continues next slide)

Checklist to avoid Pitfalls

- Use existing peer-reviewed benchmarking frameworks with built-in analysis features where possible, at least use statistical methods
- Avoid arbitrary decisions on experimental setup, including for hyperparameters (ideal: tune on different, but similar problems)
- Report complete results, including negative ones
- Verify interpretations by isolating potential causes, form new hypotheses and reflect on original expectations

Real world applications and test problems

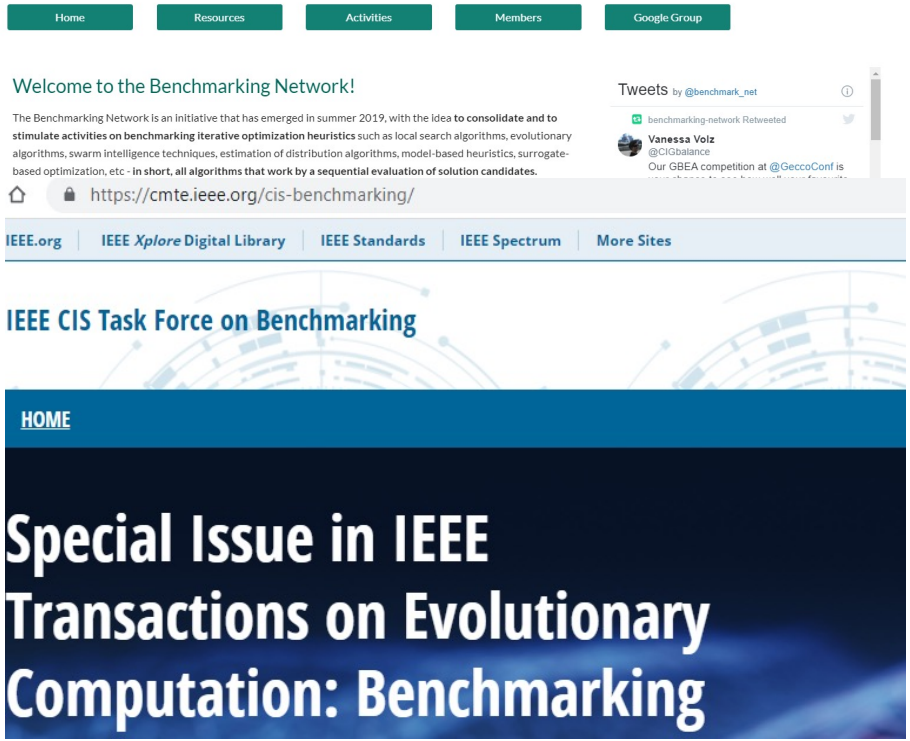
- What are the real properties / aspects of real-world, industrial, technical applications?
- Help us find out
  - Working group at
  - https://sites.google.com/view/macoda-rwp
  - Questionnaire at
  - https://docs.google.com/forms/d/e/1FAIpQLSf27nQcgJ4X690gcL6CBDcLEMUdESUYarz5_7dFFOj89U8rZQ/viewform

Summary

- Introduction
- Three examples
- Performance measurement
- Results Analysis
- Results presentation
- Reproducibility
- Pitfalls
- Summary

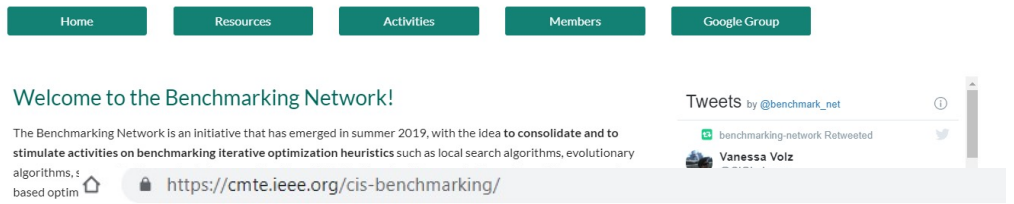
Get organized!

- Benchmarking network
  - https://sites.google.com/view/benchmarking-network/
- IEEE CIS Task Force on Benchmarking
  - https://cmte.ieee.org/cis-benchmarking/



Thank you!

Questions?



Contacts:

m.preuss@liacs.leidenuniv.nl
boris.naujoks@th-koeln.de
thomas.bartz-beielstein@th-koeln.de