ATLANTIS PRESS
Atlantis Studies in Uncertainty Modelling, volume 3
Joint Proceedings of the 19th World Congress of the International Fuzzy Systems Association (IFSA), the 12th Conference of the European Society for Fuzzy Logic and Technology (EUSFLAT), and the 11th International Summer School on Aggregation Operators (AGOP)

Measuring the Environmental Cost in the Evaluation of Metaheuristics

*__Pavel Novoa-Hernández__[a] and __Amilkar Puris__[b] and __David A. Pelta__[c]
[a]Escuela de Ciencias Empresariales, Universidad Católica del Norte, Chile. pavel.novoa@ucn.cl
[b]Facultad Ciencias de la Ingeniería, Universidad Técnica Estatal de Quevedo, Ecuador. apuris@uteq.edu.ec
[c]Department of Computer Science and A.I., Universidad de Granada, Spain. dpelta@decsai.ugr.es

Abstract

Several situations associated with the Sustainable Development Goals can be modeled as optimization problems that, under certain circumstances, can be solved with metaheuristics.

In this paper we focus in the environmental impact (carbon footprint) that running such techniques produces. Through the simulation of two typical scenarios in the context of evolutionary optimization, we aim to raise awareness about taking into account such impact when designing experiments. In our simulations we found that a) both, the characteristics of the problem and of the solver (metaheuristics) can significantly increase electricity consumption and carbon emissions; and b) running experiments in certain countries have a higher social cost than in others. We suggest some strategies for reducing the environmental impact when conducting experiments in this field.

Keywords: Metaheuristic, Carbon footprint, Computational Experiments.

1 Introduction

The Sustainable Development Goals are a universal call to action to end poverty, protect the planet and improve the lives and prospects of everyone, everywhere. The 17 Goals were adopted by all UN Member States in 2015, as part of the 2030 Agenda for Sustainable Development which set out a 15-year plan to achieve the Goals (see https://sdgs.un.org/).

At the core of such goals there are a huge number of topics related with: intelligent transport, services deployment, health promotion, smart societies, disaster and crisis management, etc. In each of these topics it is possible to recognize decision and optimization problems leading to models and frameworks (be it mathematical, linguistic, computational...) requiring suitable solution methods.

Although some of those problems do not require a technological solution, many others can be addressed using the tools provided by Artificial Intelligence (AI). The increasing computing power gave raise to an increase in both the size of the problems being tackled and the number of experiments that can be done.

Machine learning and optimization are two of the main tasks underlying the resolution of those problems and, as others are doing, we want to raise the attention on the carbon footprint that the solving procedure by itself produces (or the procedure being done to obtain the solving algorithm).

Just a few examples are needed to illustrate the current situation. In a well known report from the end of 2019 [18], authors quantified "the computational and environmental cost of training deep neural network models for natural language processing (NLP)". The simplest model took 12 hs. and emitted 26 CO_2 pounds, while the largest took the equivalent of 274,120 hs. and emitted 26,155 CO_2 pounds.

More recently, in [4] authors stated that training the GPT-3 (175B) model consumed several thousand petaflop/s-days of compute while the previous version GPT-2 (1.5B) , consumed tens of petaflop/s-days (note that the "B" in 175 and 1.5 stands for billions parameters). They ended with two relevant facts: a) the need to be cognizant of the cost and efficiency of such models, and b) that while training is computing intensive, the model usage (when trained) is cheap.

In [6], the author summarized some data and the current efforts to understand the part that artificial intelligence (AI) plays in climate change, recognizing both the role of AI techniques as problem solvers and, in

the other side of a coin, their role as significant carbon emitters.

Several authors, propose making efficiency a more common evaluation criterion for AI papers alongside accuracy and related measures and aim to add a clear statement about the carbon footprint of the research done.

In this context, the aim of this paper is to analyze the carbon footprint impact that running metaheuristics has. To the best of our knowledge, no similar studies exists. More specifically, we pose the following two research questions: 1) *which is the carbon footprint that a "standard" metaheuristics paper has*? 2) If such impact is relevant, *can we suggest a way to reduce it*?

We will design and run two "typical" experiments with metaheuristics (several methods, parameters and a benchmark) while measuring the carbon footprint indirectly via power consumption of the specific hardware and its geographical location.

2 Carbon footprint calculation

In very simple terms, the carbon footprint of a specific algorithm implementation can be estimated in two steps [2]. In the first one, an estimation of the power consumption (in kilowatt, kW) is calculated taking into account (at least) the power consumption of the CPU, the number of CPUs, the power consumption of the GPU, and the power consumption of the memory, multiplied by the running time of the algorithm. Then we obtain the kWw consumption per hour (kWh) As it is clear, this step is hardware dependant.

In the second step, we need to multiply the amount of energy by the carbon intensity, which is a measure of the grams of CO_2 emitted per kWh. In this step the source of the electricity is taken into account: if the experiment is run in a geographical region where most of the energy produced is from renewable sources, then its carbon footprint will be lower than if it is ran in a region where the energy comes from a coal-based source.

This information is readily available from `https://www.electricitymap.org`: a real-time visualisation of the Greenhouse Gas (in terms of CO_2 equivalent) footprint of electricity consumption.

Another important point is the so called Power Usage Effectiveness (PUE) value which determines how effective a data-center is at utilizing its energy. For example, a PUE of 2.0 means that for every 1 kWh of electricity that reached the server, the data center needs 2 kWh to account for waste and other services like cooling.

Several tools, that include databases of hardware specifications, are nowadays available to measure the carbon impact of computational experiments.

As indicated in [11] the emissions incurred in the training of a neural network model are related to the location of the training server and the energy grid it uses, the length of the training procedure, and the hardware on which the training takes place. They developed an emissions calculator to estimate the energy use which is available at `https://mlCO2.github.io/impact/`, and the concomitant environmental impact, of training ML models.

Another calculator is available at `https://green-algorithms.org/`. The calculator "easily integrates with computational processes as it requires minimal information and does not interfere with existing code, while also accounting for a broad range of CPUs, GPUs, cloud computing, local servers and desktop computers".

While the previous calculators provide, more or less, rough estimates of the carbon footprint, other techniques are available to provide a more precise estimate.

One example is `https://codecarbon.io/`. This Python package enables developers to track carbon dioxide (CO_2) emissions across machine learning experiments or other programs.

Another one is the *experiment-impact-tracker* [10] which is the one used in this paper and it is described below.

3 Computational experiments

In order to measure the environmental impact of running computational experiments with metaheuristics, we consider two typical scenarios in the context of evolutionary optimization: 1) single objective bound constrained optimization (SOBCO) [19] and 2) evolutionary dynamic optimization (EDO) [5, 13]. In both cases the algorithms optimized black-box functions over a continuous domains. The DEAP framework [7] was used for implementing the algorithms and problems.

For monitoring the energy consumption, carbon emissions and social cost of the experiments, we rely on the *experiment-impact-tracker* framework [10]. This open-source framework was developed in Python and is available as a Github repository[1]. Originally designed for monitoring machine learning experimentation, it can easily adapted to the context of metaheuristics. From an algorithmic perspective, Fig. 1 shows how we used the *experiment-impact-tracker* in the ex-

[1] https://github.com/Breakend/experiment-impact-tracker

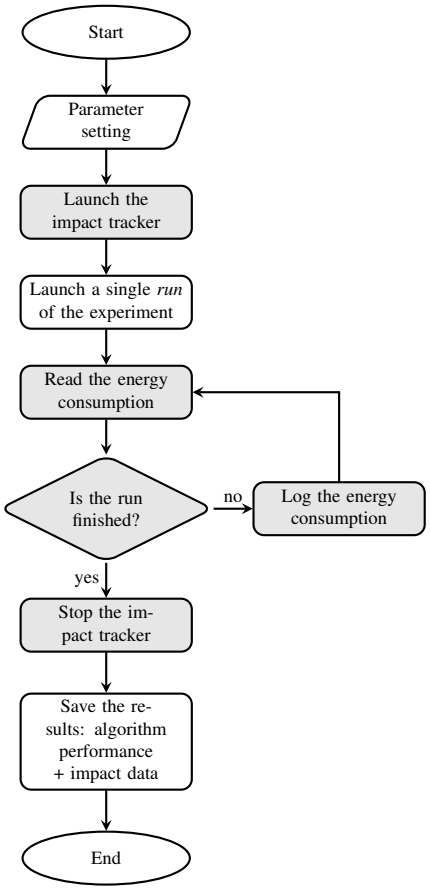

Figure 1: Flowchart of the executed code for the conducted experiments.

periments. See that we launch the impact tracker before executing a single run of the experiment (a single instance of metaheuristic vs. a single instance of an optimization problem). During this process the impact tracker reads the energy consumption from several sources (e.g. CPU, GPU, and DRAM). Further, these data are logged into a specific folder if the run is not finished. Otherwise, we stop the tracker explicitly and save the impact information and the algorithm performance into a file. This latter operation ends the program.

It is worth noting that since our program correspond to a single run, the impact and performance of the algorithm can be recorded with this granularity. In this way, at the end of the program depicted in Fig. 1, we will have the metaheuristic final performance and the impact of this run in terms of consumed energy, carbon emissions and social cost for several countries. The list of countries considered to report the social cost includes China, the United States, India, the United Kingdom, Spain, Germany, which are the top-six places where the research on metaheuristics originates[2]. We also included Chile since it was the place where the experiments presented in this paper were conducted.

We performed 30 independent runs (different random seeds) for each pair of problem and algorithm. In the case of SOBCO, the best fitness was employed as the performance measure, while for EDO, we relied on the offline error. The experiments were executed in a sequential way (one run at the time) on an iMac computer, with an Intel Core i5 (3 GHz) as CPU and 40 GB of RAM. No GPU computations were performed.

In the following subsections we provide details about both scenarios. The related source code of the experiments can be found as a GitHub project[3].

3.1 Scenario 1: Single objective bound constrained optimization

In the SOBCO scenario (Table 1), we considered problem instances derived from the combination of three objective functions and four search space dimensions. As a consequence, 12 optimization problems with different features were obtained. From the settings listed in Table 1 on these two parameters we can easily hypothesize that greatest overhead will occur in experiments involving problem instances with the highest dimension (D = 130) and objective functions *Rastrigin* or *Rosenbrock*. This is because the dimension of the search space has a multiplicative effect on the number of operations within the sums defining the objective functions, while *Rastrigin* and *Rosenbrock* involve more mathematical operations than the *Sphere* function. As a consequence of the latter, the processing units involved in running the experiment (e.g. CPU or GPU) will have more floating-point arithmetic operations to perform.

Regarding the algorithms, we rely on two popular metaheuristics: Differential Evolution (DE) [17] and the Covariance Matrix Adaptation Evolution Strategy (CMAES) [9]. Here, we did not explore different variants of these algorithms because they have by definition different complexities. For instance, we expect more overhead when running CMAES since it largely depends on the adaptation of the covariance matrix at every iteration. This translates into more floating-point arithmetic operations for the processing units.

[2]According to Scopus (http://www.scopus.com) using as the search query: TITLE-ABS-KEY (metaheuristic OR "evolutionary algorithm" OR "evolutionary computation").

[3]https://github.com/pnovoa/metaheuristics-environmental-impact

		Parameter	Settings
SOBCO	*Problems*	Problem type	Minimization
		Objective Function	$Sphere(x) = \sum_{i=1}^{D} x_i^2$
			$Rastrigin(x) = 10D + \sum_{i=1}^{D} x_i^2 - 10\cos(2\pi x_i)$
			$Rosenbrock(x) = \sum_{i=1}^{D-1} (1-x_i)^2 + 100(x_{i+1} - x_i^2)^2$
		Dimension (D)	$\in \{10, 50, 90, 130\}$
		Search space	$[-5.0, 5.0]^D$
	Algorithms	Metaheuristic	DE, CMAES
		Generations (G)	500
		Population size (λ)	$20D$
EDO	*Problems*	Problem type	Maximization
		Moving Peak Benchmark	*Scenario2*
		Peak function (PF)	$Cone(x) = \sqrt{\sum_i^D (x_i^p - x_i)^2}$
		Dimension (D)	5
		Search space	$[0.0, 100]^D$
		Number of peaks (*Peaks*)	$\in \{10, 20, 30, 40\}$
		Number of changes (*Changes*)	100
		Change severity (s)	1.0
		Change frequency (CF)	$\in \{2500, 5000, 7500, 10000\}$
	Algorithms	Metaheuristic	*mQSO*
		Number of swarms (*Swarms*)	$\in \{1, 10, 20, 30\}$
		Number of neutral particles (n)	5
		Number of quantum particles (q)	5
		Quantum radius (r_{cloud})	0.5

Table 1: Parameter settings for the experiments conducted in both scenarios: SOBCO and EDO.

3.2 Scenario 2: Evolutionary dynamic optimization

The EDO scenario (Table 1) involves experiments on the popular Moving Peaks Benchmark [12]. In our study, the problem instances were derived from the combination of two parameters that clearly affect the running time of the experiments: the number of peaks (*Peaks*) and the frequency of the changes (CF) [14]. While the former increase the number of floating-point operations related to the evaluations in the objective function, the second one makes the execution longer in terms of objective function evaluations. However, the low dimension of the problem ($D = 5$) leads us not to expect relatively low overhead in the experiments, that is, in terms of the usage of the involved processing units.

In addition, the multi-swarm quantum particle swarm optimization algorithm (mQSO) [3] was considered as the optimizer. An important parameter in mQSO is the number of sub-populations (swarms) devoted to explore the search space simultaneously. Hypothetically, increasing the number of swarms rises the computational cost of the so-called exclusion principle (devoted to avoid swarms overlapping each other). However, we do not expect too much overhead in the processing units in this case. The reason behind this hypothesis is that although the main cost in exclusion principle is the computation of the euclidean distance between the best solution from two swarms, which happens $S(S-1)/2$ times (S the number of swarms), the distance formula depends on the dimensions of the search space, which is low as noted before.

4 Results

In this section we present and discuss the results obtained. For each scenario, we compare the impact of different settings of problems and algorithms regarding the energy consumption, carbon emissions, and algorithm performance. Finally, we compare both scenarios in terms of execution time, energy consumption, CO_2 emissions and social cost.

4.1 Scenario 1: SOBCO

The energy consumed in this scenario in terms of kWh is summarized in Fig. 2-a, detailed by problem, search space dimension and metaheuristics. As expected, we observe that experiments involving the larger dimensions ($D \geq 90$) and more complex problems (*Rastrigin* or *Rosenbrock*) consume more energy. Moreover, it is clear that CMAES outperforms DE in this indicator. This is consistent with what was discussed above: the frequent adaptation of the covariance matrix involves a higher number of floating point operations.

Intuitively, more energy means more CO_2 emissions. Fig. 2-b confirms this intuition by revealing the same pattern in the data as shown in the case of energy consumption. These values of emissions are estimated for Chile, which is the country where the experiments were executed.

The algorithms' performance in terms of the best fitness (best solution found by the algorithm) is shown in Fig. 2-c. Overall, we can observe that CMAES achieved better results than DE in all problems except

for *Rastrigin* with $D = 130$. These results are consistent with those reported in the literature, see for instance [8]. However, the important fact here is that *no free lunch* exits for algorithms like CMAES: the high-quality solutions provided by CMAES come at a higher cost in energy consumption, which implies more CO_2 emissions.

4.2 Scenario 2: EDO

In this scenario, Fig. 3-a and Fig. 3-b show that experiments involving problems with the largest change frequency (CF) and number of peaks (*Peaks*) consume more energy and therefore emit more CO_2. Interestingly, in this scenario we observe lower values on these indicators if we compared with SOBCO. See for instance that the largest values in EDO are 0.0059 kWh and 0.0149 Kg CO_2, while in SOBCO 0.0349 kWh and 0.0886 Kg CO_2.

Another interesting pattern arising from Fig. 3-a and Fig. 3-b is the small effects of the number of swarms in the energy consumption and CO_2 emissions, respectively. A similar behaviour is observed in Fig. 3-c, where it is hard to find a winner variant of mQSO, that is, according to the number of swarms (x-axis).

4.3 Scenario comparison

Fig. 4 summarizes both scenarios according to their execution time, energy consumption, CO_2 emissions, and social cost. From these graphs it can be seen that although the execution time of all the experiments in SOBCO was shorter than those of EDO (Fig.4a), this scenario had higher energy consumption (Fig. 4b) and therefore higher CO_2 emissions (Fig. 4c). This apparent contradiction is a clear indication that the calculations involved in the SOBCO experiments led to increased use of the processing unit (CPU) due to the presence of high dimension problems and a metaheuristic like CMAES.

Regarding the social cost (interpreted as marginal damage), Fig. 4d shows a clear variability according to the country where the experiments could be carried out. Note, for example, that for countries like Germany, the United Kingdom, and Chile, this social cost takes negative values, while the rest of the countries have positive values. According to [16] this means that for the first three countries, running our experiments does not cause damage, but benefits. This is because their corresponding temperatures are "below the economic optimum"[16]. On the other hand, having carried out our experiments in countries like Spain, China, USA or India would have had combined social costs within the range $[0.001, 0.035]$ per tonne of CO_2.

5 Concluding remarks

Every human activity produces an impact on the environment. Although the analysis of this impact has been monitored in various contexts, much work remains to be done. In this paper we contribute to close this gap by analyzing the environmental impact of experimentation with metaheuristics. From the simulations performed we conclude that:

1. certain features of the optimization problem, (eg. dimension, objective function) are important determinants in the increase of the the experiments' energy consumption (and CO_2 emissions);
2. some historically effective metaheuristics, such as CMAES, are associated with higher energy consumption and CO_2 emissions compared to others with lower computational complexity (e.g. DE). This has important practical implications in real scenarios, because together with the performance of the metaheuristic, practitioners will have to consider another apparently conflicting objective: its environmental impact.
3. a longer execution time does not necessarily imply a higher energy consumption. Our simulations show that other factors (such as those mentioned in points 1 and 2) increase the energy consumption in experiments with low execution time;
4. if possible, it would be crucial to properly choose the location where the experiments will be run, specially for large scale experiments.
5. although the low values reported by our simulations may suggest that experimentation with metaheuristics does not significantly impact the environment, it should be kept in mind that the scenarios considered represent only a small fraction of those currently studied in this field, which involve much more complex experimental settings. The robust optimization over time [15] is a typical example.

Besides these conclusions, we suggest some actions to reduce the negative impact of experimenting with metaheuristics. First, before designing a computational experiment, one should need to understand as much as possible the theory explaining the behavior of the metaheuristic. In this way, one can estimate the degree of effectiveness of the metaheuristic and thus save unnecessary runs, which translate into less energy consumption. In this context, the implications of the *No Free Lunch* theorems [20] should be always remembered: there is no winner metaheuristic when consider all possible problems.

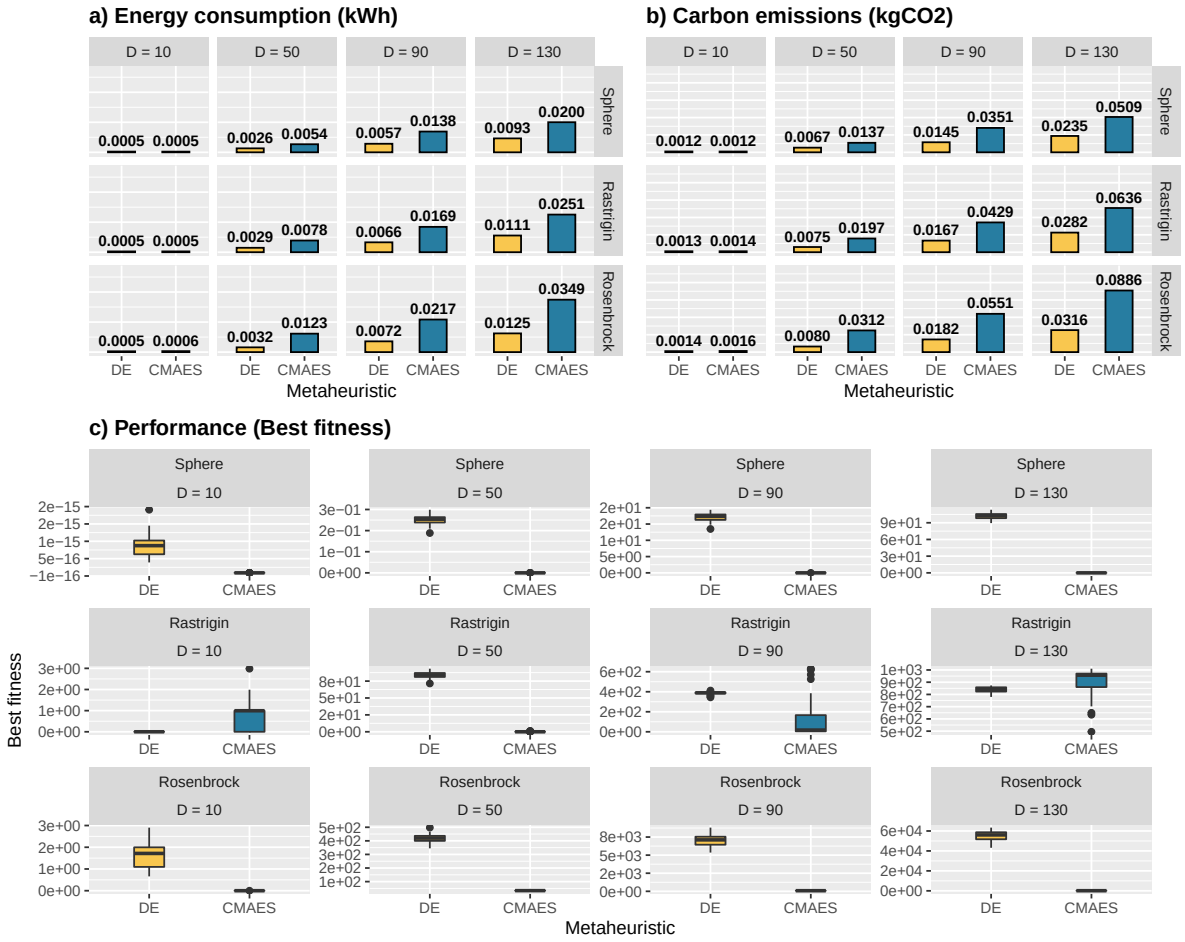

Figure 2: Energy consumption, carbon emissions and algorithm performance in SOBCO experiments.

Secondly, try some experimental designs that involve fewer combinations than full factorial designs. Literature includes important alternatives to consider [1]. A key aspect here beyond, for example, the metaheuristics parameters, is the benchmark considered. There is a tendency to increase the size of the benchmarks, and many times, this is not properly justified.

Finally, as [10] suggested, include in your papers and reports, a *Carbon statement* summarizing how your experimentation impacted in the environment. This will help raise awareness, at least in our field of research, about the rational use of computing resources.

Carbon statement

This work contributed 0.998 kg of CO_{2eq} to the atmosphere and used 0.393 kWh of electricity, having a CHL-specific social cost of carbon of $-0.0001(-0.0003, -1.1587e-05)$. The PUE multiplier is assumed to be 1.56.

Acknowledgement

D. Pelta acknowledges support from project (TIN2017-86647-P) Spanish Ministry of Economy, Industry and Competitiveness, Spain (including FEDER funds).

References

[1] J. Antoy, Design of Experiments for Engineers and Scientists: Second Edition, Elsevier, 2014.

[2] S. Bergman, How can i calculate co2eq emissions for my azure vm? (2021).
URL `http://shorturl.at/hloBN`

[3] T. Blackwell, J. Branke, Multi-swarm optimization in dynamic environments, in: Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2004, pp. 489–500.

[4] T. B. Brown, et. al, Language models are few-shot learners (2020).

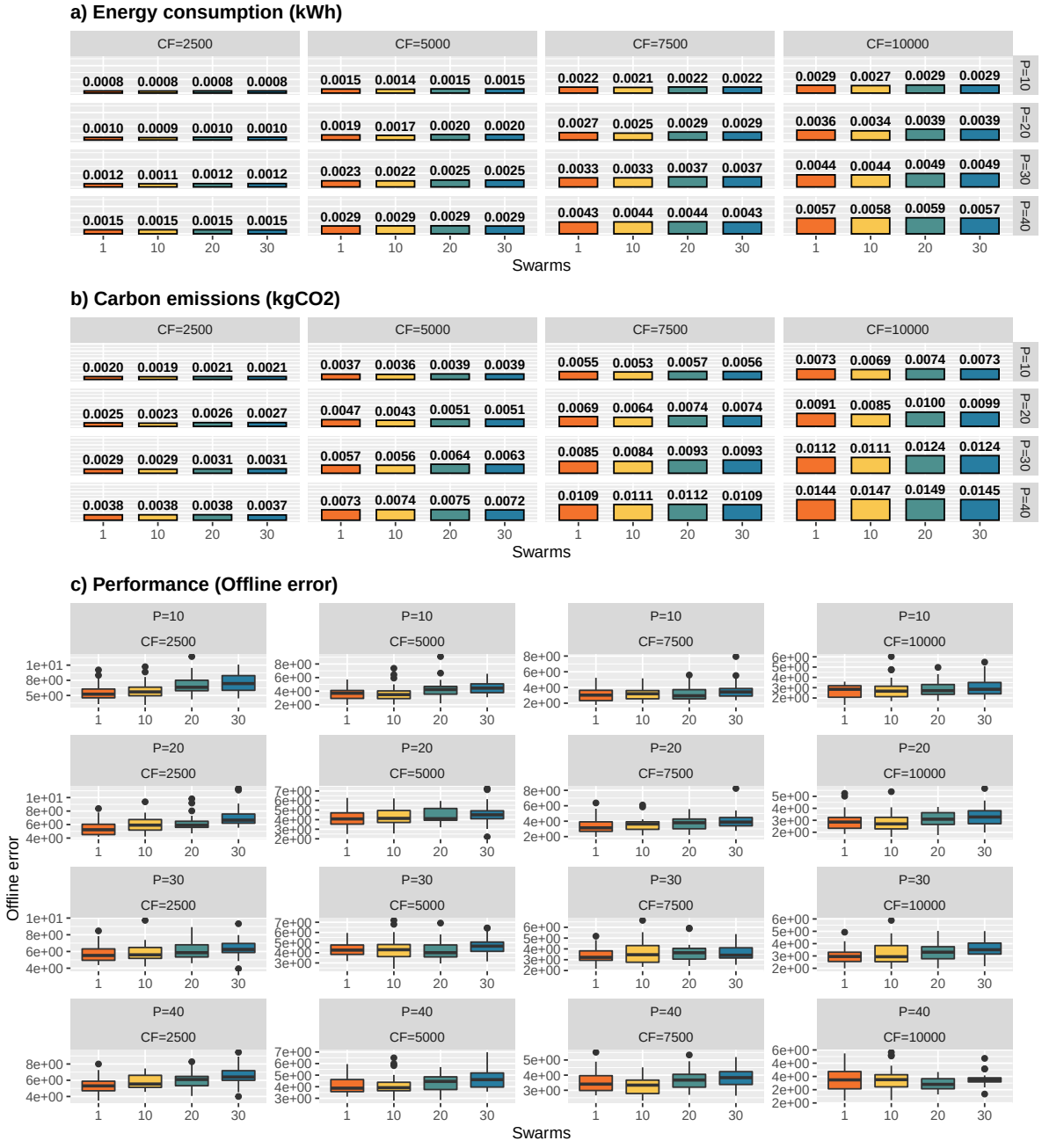

Figure 3: Energy consumption, carbon emissions and algorithm performance in EDO experiments.

[5] C. Cruz, J. R. González, D. A. Pelta, Optimization in dynamic environments: a survey on problems, methods and measures, Soft Computing 15 (7) (2010) 1427–1448.

[6] P. Dhar, The carbon impact of artificial intelligence, Nature Machine Intelligence 2 (8) (2020) 423–425.

[7] F.-A. Fortin, F.-M. D. Rainville, M.-A. Gardner, M. Parizeau, C. Gagné, Deap: Evolutionary algorithms made easy, Journal of Machine Learning Research 13 (70) (2012) 2171–2175.

[8] C. García-Martínez, P. D. Gutiérrez, D. Molina, M. Lozano, F. Herrera, Since CEC 2005 competition on real-parameter optimisation: a decade of research, progress and comparative analysis's weakness, Soft Computing 21 (19) (2017) 5573–5583.

[9] N. Hansen, The cma evolution strategy: A com-

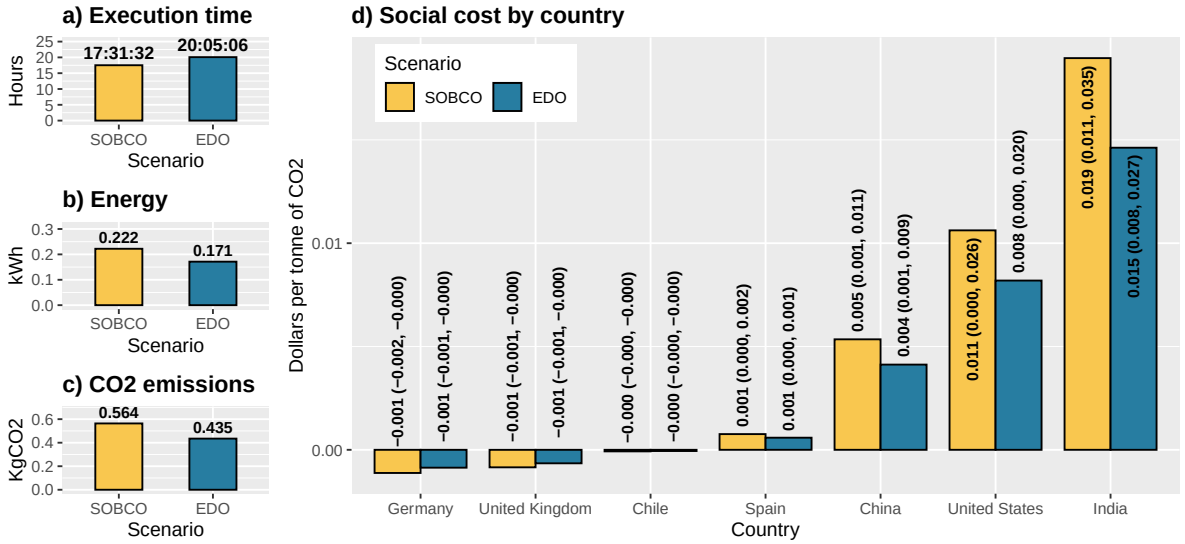

Figure 4: Summary of the scenarios in terms of execution time (a), energy consumption (b), CO_2 emissions (c), and social cost by country (d). The bars in plot (d) correspond to estimated medians (50%), while the numbers inside parenthesis to 66% confidence intervals.

paring review, in: J. A. Lozano, P. Larrañaga, I. Inza, E. Bengoetxea (Eds.), Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms, Springer Berlin Heidelberg, 2006, pp. 75–102.

[10] P. Henderson, J. Hu, J. Romoff, E. Brunskill, D. Jurafsky, J. Pineau, Towards the systematic reporting of the energy and carbon footprints of machine learning (2020).
URL https://arxiv.org/abs/2002.05651

[11] A. Lacoste, A. Luccioni, V. Schmidt, T. Dandres, Quantifying the carbon emissions of machine learning (2019).

[12] I. Moser, R. Chiong, Dynamic function optimization: The moving peaks benchmark, in: Metaheuristics for Dynamic Optimization, Springer Berlin Heidelberg, 2013, pp. 35–59.

[13] T. T. Nguyen, S. Yang, J. Branke, Evolutionary dynamic optimization: A survey of the state of the art, Swarm and Evolutionary Computation 6 (2012) 1–24.

[14] P. Novoa-Hernández, C. Cruz Corona, D. A. Pelta, A software tool for assisting experimentation in dynamic environments, Applied Computational Intelligence and Soft Computing 2015.

[15] P. Novoa-Hernández, D. A. Pelta, C. C. Corona, Approximation models in robust optimization over time - an experimental study, in: 2018 IEEE Congress on Evolutionary Computation (CEC), 2018, pp. 1–6.

[16] K. Ricke, L. Drouet, K. Caldeira, M. Tavoni, Country-level social cost of carbon, Nature Climate Change 8 (10) (2018) 895–900.

[17] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (4) (1997) 341–359.

[18] E. Strubell, A. Ganesh, A. McCallum, Energy and policy considerations for deep learning in nlp (2019).

[19] A. Wagdy, A. A. Hadi, A. K. Mohamed, P. Agrawal, A. Kumar, P. N. Suganthan, Problem definitions and evaluation criteria for the cec 2021 special session and competition on single objective bound constrained numerical optimization, Technical report, Nanyang Technological University (2020).
URL http://home.elka.pw.edu.pl/~ewarchul/cec2021-specification.pdf

[20] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 67–82.