Statistical Computing in R
Report

Implementation of NMMSO in R

Submitted by

425699 Daniel Carriola
425599 Jannik Hoffjann

Under the guidance of
Dr. Mike Preuss

Information Systems and Statistics
ERCIS
Münster - NRW - Germany

Winter Semester 2015/16

Contents

1 Introduction

In the recent years R has become the statistical programming language of choice for many scientist. The strength of R of being a domain specific language has also become one of its weaknesses. Since new research findings in statistical computing are split up over several languages like R, Matlab or SciPy[1] it often becomes difficult to compare new methods with established ones. Since it is also hard to interface those languages due to different architectures, data storage mechanisms there is often no other way than to reimplement new methods in a different programming language to create a common scope.

An example for a well perceived new finding in statistical computing is the NMMSO-Algorithm by Jonathan E. Fieldsend (Fieldsend 2014). It won the niching competition in 2015 held by the CEC and is only written in Matlab. Since the chair 'Information Systems and Statistics' at the Westfälische Wilhelms-Universität Münster, Germany is mainly concentrating its work on Statistical Computing in R an implementation of this algorithm became interesting.

As part of this Seminar Project in the context of the Seminar 'Statistical Computing in R' a reimplementation of the NMMSO algorithm in R (nmmso.R) will be presented. During this technical documentation, the general function of the algorithm and the used test cases by the CEC will be shown. Afterwards the structure and used techniques and libraries, as well as problems and pitfalls due to the different behaviours of R and Matlab, will be shown. The documentation will be closed by the benchmarking results and different test cases.

It was the goal of this project to keep up high comparability with the original code, to ensure the correct functionality and easily implement changes to the original codebase in this program. To reach this, unit tests were used where possible and continouous comparison between part results of the original implementation and nmmso.R where used to ensure functioning. Additionally a benchmarking suite, which builds on the CEC Benchmarking Suite for Niching Algorithm was implemented to evaluate and test the functioning of nmmso.R with the same characteristics as in the original implementation.

2 General Function

The starting point of the project was the paper provided by Dr. Jonathen E. Fieldsend (Fieldsend 2014) on the Niching Migratory Multi-Swarm Optimiser (NMMSO) algorithm. NMMSO is a multi-modal optimiser which relies heavily on multiple swarms which are generated on the landscape of an function in order to find the global optima. It is build around three main pillars: (1) dynamic in the numbers of dimensions, (2) self-adaptive without any special preparation and (3) exploitative local search to quickly find peak estimates (Fieldsend 2014, p. 1).

Multi-modal optimisation in general is not that different from well known and widely discussed single-objective optimisation, but in difference to it the goal of the algorithms in the multi-modal is not to find just one single optimising point but all possible points (Fieldsend

[1] SciPy is a common library for the Python Programming language which brings Statistical Computing capabilities to the language.

2014, p. 1). To reach this goal many multi-modal optimization algorithms use strategies oriented on the biological world and utilize swarm intelligence to find optima defined by the search parameters (Yang 2009). In order to do so, many early multi-modal optimisation algorithms needed defined parameters (Fieldsend 2014, p. 1).

Newer algorithms fall in the field of self-tuning and try to use different mathematical paradigms like nearest-best clustering with covariance matrices (Preuss 2012) and strategies like storing the so far best found global optima estimators to provide them as parameters for new optimisation runs (Epitropakis et al. 2013). Contradictory to that NMMSO goes another way and uses the the swarm strategy in order to find which store their current (Fieldsend 2014).

In order to do so NMMSO follow a strict structure which can be seen in the following pseudo-code

```
nmmso(max_evals, tol, n, max_inc, c_1, c_2, omega)
    S: initialise_swarm(1)
    evaluations := 1
    while evaluations < max_evals:
        while flagged_swarms(S) == true:
            {S, m} := attempt_merge(S, n, tol)
            evals := evals + m
        S := increment(S, n, max_inc, c_1, c_2, omega)
        evals := evals + min(|S|, max_inc)
        {S, k} := attempt_separation(S, tol)
        evals := evals + k
        S := add_new_swarm(S)
        evals := evals + 1
    {X*, Y*} := extract_gbest(S)
    return X*,Y*
```

This structure wasn't modified during the reimplementation of NMMSO to keep comparability and the possibility to fix bugs at a high level. The only newly introduced setting was the possibility to modify the c_1, c_2, w as parameters from the outside. In the original version those parameters are part of the program code.

	standard value	used value
evaluations	0	0
max_evol	100	100
tol_val	10^-6	10^-6
c_1	2.0	2.0
c_2	2.0	2.0
omega	0.1	0.1

3 CEC Algorithms

3.1 CEC

The IEEE Congress of Evolutionary Computation (CEC) is one of the largest, most important and recognised conferences within Evolutionary Computation (EC). It is organised by the IEEE Computational Intelligence Society in cooperation with the Evolutionary Programming Society, and covers most of the subtopics of the EC.

In order to validate the potential of the NMMSO algorithm, it was submitted to the IEEE CEC 2013 held in Cancun, Mexico. Here, Dr. Fieldsend was provided with some multimodal benchmark test functions with different dimension sizes and characteristics, for evaluating niching algorithms developed by Dr. Xiaodong Li, Dr. Andries Engelbrecht and Dr. Michael G. Epitropakis (Epitropakis et al. 2013). They state that even if several niching methods have been around for many years, further advances in this area have been hindered by several obstacles; most of the studies focus on very low dimensional multimodal problems (2 or 3 dimensions) making this more complicated to asses theses methods' scalability to high dimensions with better performance. The benchmark tool includes 20 test functions (in some cases the same function but with different dimension sizes), which includes 10 simple, well-known and widely used benchmark functions, based on recent studies, and more complex functions following the paradigm of composition functions. In the following section, they will be briefly explained:

- `F1: Five-Uneven-Peak Trap (1D)`
- `F2: Equal Maxima (1D)`
- `F3: Uneven Decreasing Maxima (1D)`
- `F4: Himmelblau (2D)`
- `F5: Six-Hump Camel Back (2D)`
- `F6: Shubert (2D, 3D)`
- `F7: Vincent (2D, 3D)`
- `F8: Modified Rastrigin - All Global Optima (2D)`
- `F9: Composition Function 1 (2D)`
- `F10: Composition Function 2 (2D)`
- `F11: Composition Function 3 (2D, 3D, 5D, 10D)`
- `F12: Composition Function 4 (3D, 5D, 10D, 20D)`

All of the test functions are formulated as maximisation problems. F1, F2 and F3 are simple 1D multimodal functions, while F4 and F5 are simple 2D functions and not scalable. F6 to F8 are scalable multimodal functions. The number of global optima for F6 and F7 are determined by the dimension. However, for F8, the number of global optima is independent from the dimension, therefore it can be controlled by the user. F9 to F12 are scalable multimodal functions constructed by several basic functions with different properties (Sphere function, Grienwank, Rastrigin, Weierstrass and the Expanded Griewank's plus Rosenbrock's function). F9 and F10 are separable, and non-symmetric, while F11 and F12 are non-separable, non-symmetric complex multimodal functions. The number of global optima in all of the composition functions is independent from the number of dimensions, therefore can be controlled by the user (Epitropakis et al. 2013).

Maybe write each math equation or the R code

https://en.wikipedia.org/wiki/IEEE_Congress_on_Evolutionary_Computation

3.2 Implementation and Pitfalls

write also about the count_goptima and so on

---

4 The Implementation

4.1 Structure of the project

After analysing the algorithm provided in Matlab by Dr. Fieldsend, it was decided to first translate each of the functions into the R programming language. At first instance, this task seemed to be simple because most of the functions were basically managing matrices and vectors, but later this became a problem that will be addressed in the pitfalls' section of this paper.

Once all the NMMSO functions existed in R and having the input data, the testing phase started. It has be said, that one of the biggest problems when you code an already existing program into another programming language, is the different behaviours corresponding to each object (in case of an object-oriented language) or its main structure. The first runs came with several errors regarding the matrix generation and handling, slowing down the project in a near future. Using GitHub, it was easier to attack these problems in parallel, having one developer reviewing different functions and the other one, fixing other bugs and continue the testing phase. Also, this was achieved in an easier way, thanks that each function was coded in an independent R file, making easier and faster the debugging and the fixing of each problem.

During the developing time, an issue raised with the CEC benchmark tool. In order to compare the R implementation of the NMMSO algorithm with the original one, it was mandatory to use this tool to test each of its functions with the new algorithm and compare results. After several complications with the original test suite (these complications will be addressed in the pitfalls' section), it was decided to recode each of the functions as an independent R package to avoid any further complication and having an easier and more trustworthy comparison of the NMMSO algorithm in R.

4.2 Pitfalls and Problems

test

5 Benchmark and Comparison

To compare nmmso.R with the original NMMSO the CEC test cases were used to run the same benchmarks as in the original submission (Fieldsend 2014). There 4 different Ratios were used to measure the performance of certain algorithms. Three of those measures (Peak Ratio, Success Ratio and Convergence Speed) have been introduced in (Epitropakis et al. 2013, pp. 6–7) to create a common point of comparison. The fourth ratio is special for the nmmso algorithm since it tracks the number of swarms over the iterations of the algorithm. Nmmso.R uses the same measures to reach the highest comparability possible.

The first measure used is the Success Ratio (SR). The Success Ratio is defined as the percentage of Successful runs (runs that found all global optima) over all runs (Li et al. 2013, p. 7). As for the other ratios this measure was taken over several independent runs and collectively evaluated. The taken measures for the Success Ratio can be found in Table 2.

$$\frac{successful\ runs}{NR} = SR$$

Here NR denotes the Number of runs done to reach this measure.

Table 2: Success Ratio over given runs (Measure of share of runs which found all global optima)

	0.1	0.01	0.001	0.0001	0.00001	runs
F1	1	1	1	1	1	37
F2	1	1	1	1	1	35
F3	1	1	1	1	1	37
F4	1	1	1	1	1	37
F5	1	1	1	1	1	34
F6	1	1	1	1	0	34
F7	1	1	1	1	1	34
F8	1	1	1	0.94	0.71	17
F9	0.95	0.95	0.95	0.95	0.95	20
F10	1	1	1	1	1	34
F11	1	1	1	1	1	33
F12	1	1	1	1	1	34
F13	1	1	1	1	1	34
F14	1	1	1	1	1	33
F15	0.96	0.96	0.96	0.96	0.96	26
F16	0	0	0	0	0	13
F17	0.17	0	0	0	0	12
F18	0.38	0.38	0.38	0.31	0.31	16
F19	0	0	0	0	0	13
F20	0	0	0	0	0	12

The second measure introduced by the CEC committee and also used by Dr. Fieldsend is

the Convergence Rate. The Convergence Rate (CR) measures the needed evaluations per Accuracy and Function to find all global optima (Li et al. 2013, p. 7). This measure takes the mean of evaluations over all runs. The results of this measure can be found in Table 3.

$$\frac{\sum_{n=1}^{NR} evals_n}{NR} = CR$$

In this measure *evals* denotes the number of evaluations done.

Table 3: Convergence Rates over given runs (Mean of evaluations needed to find all global optima, if all optima have never been found the maximum allowed evaluations for that function were taken.)

	0.1	0.01	0.001	0.0001	0.00001	runs
F1	622	815	1018	1205	1441	37
F2	179	269	397	533	640	35
F3	35	170	273	390	513	37
F4	506	735	961	1201	1455	37
F5	82	194	321	522	788	34
F6	19519	24388	30499	42404	200001	34
F7	8564	9211	10604	12328	14646	34
F8	194031	231763	271058	320284	354533	17
F9	185670	189360	204110	219559	228586	20
F10	887	1326	1728	2254	2758	34
F11	3692	5747	7347	8549	9164	33
F12	17321	25823	38464	44660	51792	34
F13	11338	15676	19107	23152	26802	34
F14	28776	34298	48738	59775	68576	33
F15	107225	129151	149266	172770	189144	26
F16	400001	400001	400001	400001	400001	13
F17	377470	400001	400001	400001	400001	12
F18	299577	302391	304385	309050	309450	16
F19	400001	400001	400001	400001	400001	13
F20	400001	400001	400001	400001	400001	12

The third measure is the Peak Ratio (PR). It measures the share of found global optima over all runs (Li et al. 2013, p. 7). The results of this evaluation can be found in Table 4.

$$\frac{\sum_{n=1}^{NR} NOF_n}{NKO * NR} = PR$$

In this measure NOF denotes the number of found optima per run and NKO the number of known optima for the function.

Table 4: Peak Ratio over given runs (Share of found global optima over all runs)

	0.1	0.01	0.001	0.0001	0.00001	runs
F1	1	1	1	1	1	37
F2	1	1	1	1	1	35
F3	1	1	1	1	1	37
F4	1	1	1	1	1	37
F5	1	1	1	1	1	34
F6	1	1	1	1	0	34
F7	1	1	1	1	1	34
F8	1	1	1	1	0.98	17
F9	1	1	1	1	1	20
F10	1	1	1	1	1	34
F11	1	1	1	1	1	33
F12	1	1	1	1	1	34
F13	1	1	1	1	1	34
F14	1	1	1	1	1	33
F15	1	1	1	1	1	26
F16	0.01	0	0	0	0	13
F17	0.77	0.74	0.72	0.71	0.67	12
F18	0.83	0.83	0.83	0.8	0.8	16
F19	0.44	0.44	0.42	0.42	0.4	13
F20	0.15	0.14	0.14	0.14	0.12	12

As a fourth measure, which wasn't introduced by the CEC committee, but used in the original nmmso implementation (Fieldsend 2014) the Number of Swarms was chosen. Since this is a continuous measure and therefore no calculation is needed this measure is pictured as graphs. The graphs can be found in Figure 1. The show the development of $numberofswarms$ kept by nmmso.R over all iterations. Important to notice here is that *iterations* is different from the *evaluations* referenced in the other measures. Iterations are calls to start single runs of nmmso.R and is therefore different from the evaluations taken within the program.

Additionally a fifth measure was introduced which denotes the runtime of nmmso.R for the single functions. These times were taken on the ZIVHPC a scientific High Perfomance Computing Cluster by Westfälische Wilhelms-Universität Münster. Since the nmmso.R is a strictly sequential algorithm the runtimes for single runs will comparable on common computers. The ZIVHPC was only used to parallelize the single runs.

When comparing those measures with the ones given in the original paper (Fieldsend 2014)

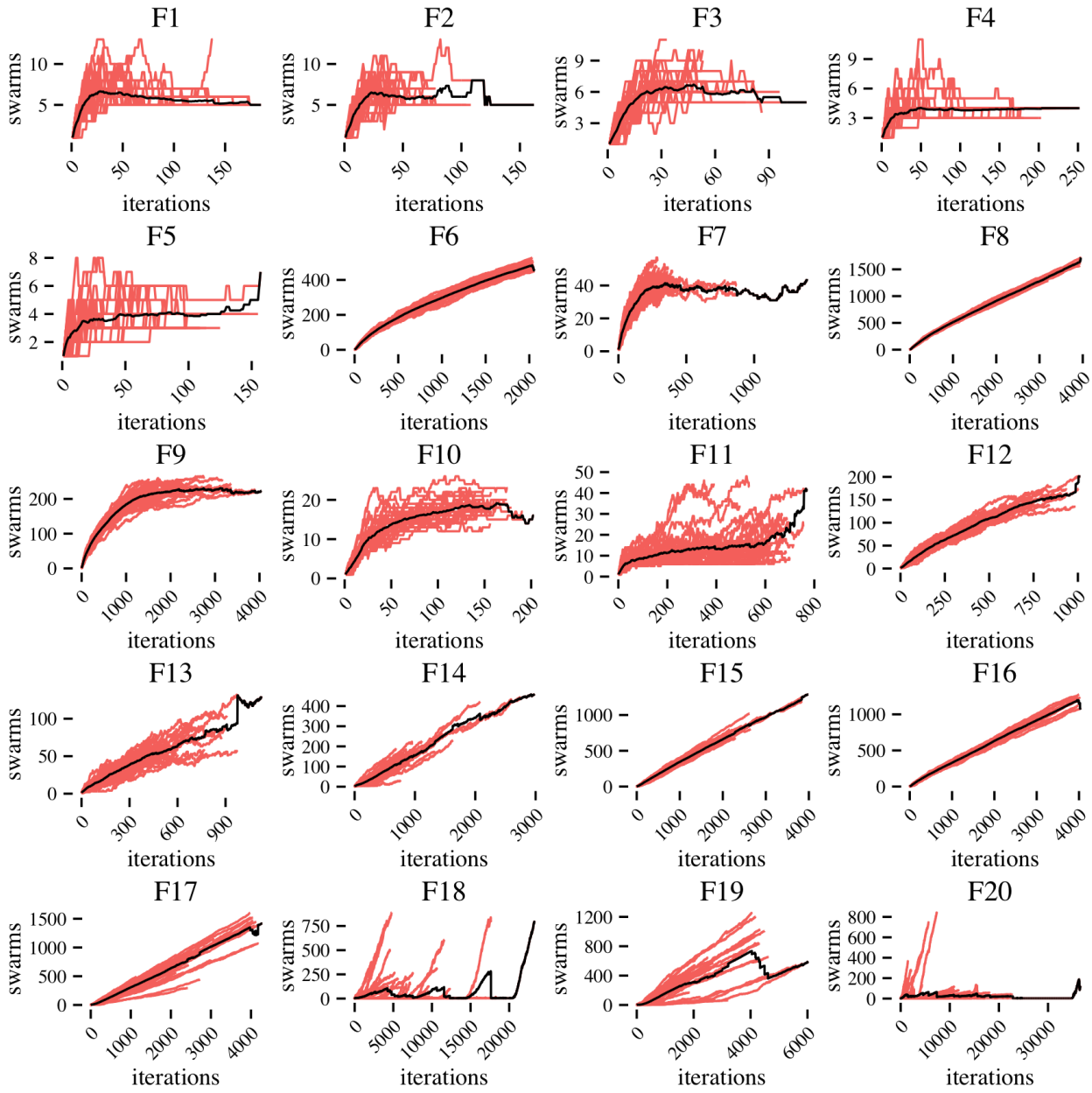

Figure 1: plot of chunk trend curve of kept swarms over all 20 functions. The red curves show the number of swarms kept for each single run. The black line shows the mean of kept swarms over these runs.

it can be seen that the reimplementation nmmso.R is an overall good resemblance of the original algorithm. The three CEC measures are close to the original taken measures and the trend curves for the number of kept swarms have similar trends.

The biggest differences between the benchmarking results of the two implementations can be seen in the general results of function 14, 15, 16 and 18, as well as in the number of created swarms for the n-dimensional functions:

(1) Function 14 and 15 have a *Success Ratio* of 1 aswell as as *Peak Ratio* of one 1 for all accuracy levels. Additionally nmmso.R sometimes found all global optima for Function 18. In contradiction to that the evaluation of all thre function almost never result in the finding of global optima in the evaluation of the original implementation. Only at the lowest accuracy the original implementation is able to find all global optima for Function 14 (Fieldsend 2014, p. 16). It is hard to say if this difference is equal to an error in the implementation of nmmso.R or if an error in the original implementation was fixed. Also this could be a difference in the reimplementation of the CEC Benchmarking Tool. Nevertheless, this is interesting point of discussion and worth evaluating.

(2) nmmso.R performs noticeable worse for Function 16 than the original function. While nmmso.R has a *Peak Ratio* of 0.01 for an accuracy of 0.1 and 0 for all others, the original implementation reaches a *Peak Ratio* of around 0.6 for all accuracies. This might be to an implementation error in the CEC Benchmarking Tool. Since it is so significantly worse that it is unlikely that this difference would only occur in one test function.

(3) Almost all algorithm runs on high-dimensional functions (F12-F20) result in a high number of swarms, while all other results regarding this functions are comparable to the original results. This difference becomes very clear in the case of Functions 17-20. In the paper addressing the original paper the x-axis rank from 0-40,000 iterations, while for the reimplementation limit of 4,000 for Function 17, of 20,000 for Function 18, 6,000 for Function 19 and of 30,000 for Function 20 is enough to show all data sets. This is connected to the creation of much more swarms, which leads to an earlier depletion of the maximum allowed number of evaluations.

6 Conclusion

test

7 Acknowledgements

We want to thank Dr. Jonathan Fieldsend for his continuous help via mail during this seminar. Also the committee of the CEC was always available for questions and concerns during our work. Furthermore a special thanks goes to all employees of the chair for 'Information Systems and Statistics' including Dr. Mike Preuß, Jakob Bossek and Pascal Kerschke who were available for any questions regarding the implementation and this report at all times.

Epitropakis, M. G., Li, X., and Burke, E. K. 2013. "A dynamic archive niching differential evolution algorithm for multimodal optimization," in *Evolutionary computation (cEC), 2013 iEEE congress on*, IEEE, pp. 79–86.

Fieldsend, J. E. 2014. "Running up those hills: Multi-modal search with the niching migratory multi-swarm optimiser," in *Evolutionary computation (cEC), 2014 iEEE congress on*, IEEE, pp. 2593–2600.

Li, X., Engelbrecht, A., and Epitropakis, M. G. 2013. "Benchmark functions for cEC'2013 special session and competition on niching methods for multimodal function optimization," *RMIT University, Evolutionary Computation and Machine Learning Group, Australia, Tech. Rep.*

Preuss, M. 2012. "Improved topological niching for real-valued global optimization," in *Applications of evolutionary computation*, Springer, pp. 386–395.

Yang, X.-S. 2009. "Firefly algorithms for multimodal optimization," in *Stochastic algorithms: Foundations and applications*, Springer, pp. 169–178.