Statistical Computing in R

Report

Implementation of NMMSO in R

Submitted by

425699 Jannik Hoffjann
425699 Daniel Carriola

Under the guidance of
Dr. Mike Preuss

Westfälische Wilhelms-Universität Münster

Information Systems and Statistics
ERCIS
Münster - NRW - Germany

Winter Semester 2015/16

Contents

List of Figures

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 new well perceived new finding in statistical computing is the NMMSO-Algorithm by Jonathan E. Fieldsend [1]. 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 Wilhems-Universität Münster 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 will be presented. During this technical documentation the general function of the algorithm will and the used test cases by the CEC will be shown. Afterwards the structure and used techniques and libraries, aswell as problems and pitfalls due to the different behaviors of R and Matlab will be shown. The documentation will be closed by the benchmarking results and different test cases.

... write a bit more here.

---

2 The Algorithm

2.1 General Function

Starting point of the project was the paper provided by Jonathen E. Fieldsend [1] on the Niching Migratoy 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 algorithm in order to find the global optimum. 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 [1].

Multi-modal optimization 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 optimizing point but all possible points [1]. In order to do so, many early multi-modal optimization algorithms needed highly defined parameters [TODO: quote needed].

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

Newer algorithms fall in the field of self-tuning and try to use different mathematical paradigms like nearest-best clustering with covariance matrices [2] and strategies like storing the so far best found global optima estimators to provide them as parameters for new optimization runs [3]. Contradictory to that NMMSO goes another way and uses the the swarm strategy in order to find which store their current [1]

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, chi, w)
    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, chi, w)
        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_gebsest(S)
    return X*,Y*
```

2.2 CEC

test

---

3 The Implementation

3.1 Structure of the project

test [#fieldsend_2014]

3.2 Pitfalls and Problems

test

3.3 Benchmark and Comparison

```
## Loading required package: lattice
## Loading required package: MASS
##
## Attaching package: 'memisc'
##
## The following object is masked from 'package:plyr':
##
##     rename
##
## The following objects are masked from 'package:stats':
##
##     contr.sum, contr.treatment, contrasts
##
## The following objects are masked from 'package:base':
##
##     as.array, trimws
```

0.1	0.01	0.001	0.0001	0.00001
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9
10	10	10	10	10
11	11	11	11	11
12	12	12	12	12
13	13	13	13	13
14	14	14	14	14
15	15	15	15	15
16	16	16	16	16
17	17	17	17	17
18	18	18	18	18
19	19	19	19	19
20	20	20	20	20

```
## Warning in file(filename, "r", encoding = encoding): cannot open file 'R/
## cec_2015_problem_data.R': No such file or directory
```

```
## Error in file(filename, "r", encoding = encoding): cannot open the connection
```

```
## Error in eval(expr, envir, enclos): could not find function "grid.arrange"
```

3.4 Testing and alternative parameter settings

test

4 Discussion

test

5 Conclusion

test

[1] J.E. Fieldsend, Running up those hills: Multi-modal search with the niching migratory multi-swarm optimiser, in: Evolutionary Computation (CEC), 2014 IEEE Congress on, IEEE, 2014: pp. 2593–2600.

[2] M. Preuss, Niching the cMA-eS via nearest-better clustering, in: Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation, ACM, New York, NY, USA, 2010: pp. 1711–1718. doi:10.1145/1830761.1830793.

[3] M.G. Epitropakis, X. Li, E.K. Burke, A dynamic archive niching differential evolution algorithm for multimodal optimization, in: Evolutionary Computation (CEC), 2013 IEEE Congress on, IEEE, 2013: pp. 79–86.