Proceedings of the 11th International Conference
on Computational and Mathematical Methods
in Science and Engineering, CMMSE2011
Beenidorm, Spain, 26-30 June 2011

Solving Nonlinear Equations by a Tabu Search Strategy

Gisela C.V. Ramadas[1] and Edite M.G.P. Fernandes[2]

[1] *Department of Mathematics, Polytechnic Institute of Porto, 4200-072 Porto, Portugal*
[2] *Algoritmi R&D Center, University of Minho, 4710-057 Braga, Portugal*
emails: gcv@isep.ipp.pt, emgpf@dps.uminho.pt

Abstract

Solving systems of nonlinear equations is a problem of particular importance since they emerge through the mathematical modeling of real problems that arise naturally in many branches of engineering and in the physical sciences. The problem can be naturally reformulated as a global optimization problem. In this paper, we show that a metaheuristic, called Directed Tabu Search (DTS) [16], is able to converge to the solutions of a set of problems for which the *fsolve* function of MATLAB® failed to converge. We also show the effect of the dimension of the problem in the performance of the DTS.

Key words: nonlinear equations, metaheuristic, tabu search

1 Introduction

The numerical solution of some problems in engineering, chemistry, physics, medicine and economic areas, aims at determining the roots of a nonlinear system of equations. The modeling of these problems can lead to simple and almost linear systems, but mostly large, complex and difficult to solve systems of nonlinear equations arise. In this paper we consider solving the nonlinear system of equations

$$F(x)=0 \tag{1}$$

where

$$F:\Omega\subset\Re^n\to\Re^n,\quad \Omega\equiv[l,u]=\left\{x\in\Re^n : l_i\le x_i\le u_i, i=1,...,n\right\},$$
$$F(x)=\left(f_1(x),...,f_n(x)\right)^T$$

and the functions

$$f_i(x),\quad i=1,...,n$$

are continuously differentiable, using a metaheuristic. Probably the most famous techniques are based on Newton's method [2,6,8,12,24,29]. They require analytical or numerical first derivative information. Newton's method is the most widely used algorithm for solving nonlinear systems of equations. It is computationally expensive, in particular if n is large, since a system of linear equations is required to be solved at each iteration. The Quasi-Newton methods use less expensive iterations than Newton, but their convergence properties are not very different. In general, Quasi-Newton methods avoid either the necessity of computing derivatives, or the necessity of solving a full linear system per iteration or both tasks [25]. In [13], a new technique for solving systems of nonlinear equations reshaping the system as a multiobjective optimization problem in proposed. The authors applied a technique of evolutionary computation to solve the problem obtained after the change. In [14], the authors propose techniques for computing all the multiple solutions in nonlinear systems. Another technique to solve systems of nonlinear equations is presented in [18], where a heuristic continuous global optimization GRASP is applied. A genetic algorithm is proposed in [4].

The problem of solving a nonlinear system of equations can be naturally formulated as a global optimization problem. Problem (1) is equivalent, in the sense that it has the same solution, to finding the globally smallest value of the l_2-norm error function, related to solving the system of equations (1), defined by

$$\min_{x\in\Omega\subset\Re^n} \Psi(x) \equiv \left\|F(x)\right\|_2 . \tag{2}$$

Here, the global minimum, and not just a local minimum, of the objective function $\Psi(x)$, in the set Ω, is to be found. The classical local search methods, like Newton-type methods, have some disadvantages, when compared to global search methods. In particular

i) the final solution is heavily dependent on the the starting point of the iterative process;
ii) they can be trapped in local minima;
iii) they require differentiable properties of all the equations of nonlinear system.

We use the Example 1 below to show this local trap behavior.

Example 1: Consider the following system of nonlinear equations

$$f_1(x_1, x_2) = x_1 - \sin(2x_1 + 3x_2) - \cos(3x_1 - 5x_2) = 0$$
$$f_2(x_1, x_2) = x_2 - \sin(x_1 - 2x_2) + \cos(x_1 + 3x_2) = 0.$$

Figure 1 shows the graphical representation of the l_2-norm error function $\Psi(x)$. The multi-modal nature of $\Psi(x)$ makes the process of detecting a global minimum a difficult one. Nine different starting points were used to solve Example 1 by *fsolve* from MATLAB®. In this MATLAB function, the default trust-region dogleg algorithm with no analytical Jacobian is used. Although all

the nine starting points are in the neighborhood of the solution, the method converges to the required solution only twice.

Figure 1. Graphical representation of $\Psi(x)$ for Example 1.

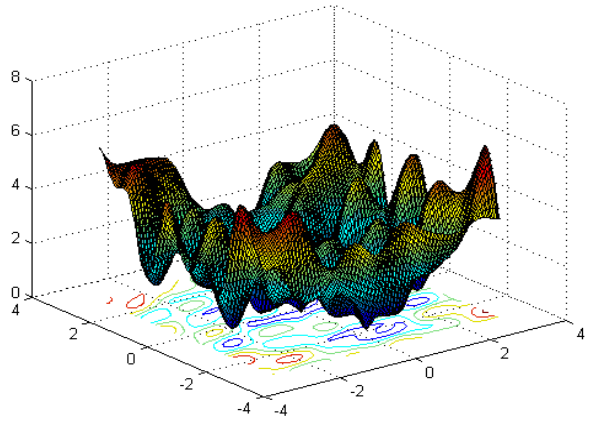

Table 1 shows the results obtained from MATLAB. The first column in the table presents the tested starting points, $x^{(0)}$, as well as the value of the output parameter "exitflag" of MATLAB. The value "1" means that the method converged to a root where the first-order optimality measure is less than a pre-specified tolerance, and "-2" means that it converged to a point which is not a root where the sum-of-squares of the function values is greater than or equal to a pre-specified tolerance.

Table 1. Solutions obtained by *fsolve* from MATLAB for different starting points.

Starting point $x^{(0)}$ /"exitflag"	(f_1, f_2) at solution	n. of iterations	n. of function evaluations
(0, 0) / "1"	(-0.41e-12, -0.24e-12)	5	18
(1, 1) / "1"	(0.36e-8, 0.23e-8)	6	21
(0, 1) / "-2"	(-0.03, 0.94)	27	60
(2, 2) / "-2"	(0.11, 0.88)	31	70
(-1, 1) / "-2"	(-0.55, 0.65)	55	130
(1, -1) / "-2"	(0.01, -0.04)	24	57
(-1, -1) / "-2"	(-0.52, 0.10)	31	74
(2, -2) / "-2"	(-0.16, -0.57)	32	71
(-2, -2) / "-2"	(-0.17, -1.53)	34	77

Thus, to be able to converge to a global solution, a global search strategy is required. The most important global search techniques invoke exploration and exploitation search procedures aiming at:

i) diversifying the search in all the search space;
ii) intensifying the search in promising areas of the search space.

A well-known class of global search techniques, the metaheuristics, use random procedures that invoke artificial intelligence tools and simulate nature

behaviors. The word “metaheuristics” is used to describe all heuristics methods that are able to achieve a good quality solution in a reasonable time. Due to their random features, metaheuristics have, in general, slow convergence since they may fail to detect promising search directions in the neighborhood of a global minimum.

There are two classes of metaheuristics. A population-based heuristic defines and maintains throughout the iterative process a set of solutions. The most known population-based heuristic is the Genetic Algorithm [10]. A point-to-point heuristic defines just one solution at the end of each iteration which will be used to start the next iteration. Simulated Annealing (SA) [15] and Tabu Search [9] are two examples of point-to-point methods.

The Tabu Search (TS) is a metaheuristic developed primary for solving combinatorial problems [9]. The TS introduced by Cvijović and Klinowski [5] for continuous optimization guides the local search out of local optima and has the ability to explore new regions. It is an iterative procedure that maintains a list of the movements most recently made, avoiding in subsequent iterations the execution of movements that lead to solutions already known to have been visited. Usually, the slow convergence of TS is overcame by incorporating a classical local search strategy into the main algorithm. In general, this type of hybridization occurs in the final stage of the iterative process when the solution is in the vicinity of the global solution. An example of such method is presented in [16]. The therein proposed method, called Directed Tabu Search (DTS), uses strategies, like the Nelder-Mead method [28] and the Adaptive Pattern Search [15], to direct a tabu search.

This paper aims at assessing the performance of a tabu search method when solving a system of nonlinear equation (1), using the function $\Psi(x)$ as a measure of the progress of the algorithm towards the solution. According to the formulation (2), this means that the fitness of each trial solution x is assessed by evaluating the function Ψ at x. And a solution $\tilde{x}$ is better than x if $\Psi(\tilde{x}) < \Psi(x)$. In this paper, and due to the reported success when solving global optimization problems of the form (2), the DTS variant of the tabu search is extended to be able to solve nonlinear systems of equations. In particular, we aim at analyzing the behavior of the extended version of the DTS method when solving some difficult problems that are not solved by Newton-type methods.

The organization of the paper is as follows. Section 2 describes the Directed Tabu Search method and the Section 3 reports the computational experiments. Finally, we summarize our conclusions in Section 4.

2 The Metaheuristic Tabu Search

2.1 Basic Tabu Search

TS is an iterative process which operates in the following way. The algorithm starts with a randomly generated initial solution, x, and by applying pre-defined

moves in its neighborhood it generates a set Y of solutions. The objective function to be minimized is evaluated at all solutions in Y, and the best of all, y, becomes the current solution, $x \leftarrow y$ (even if it is worse than x). Accepting uphill moves, the algorithm avoids to get trapped in a local mínimum. The previous procedure is repeated until a given stopping condition is reached. Further, the algorithm also stops when the solution does not improve for *Nmax* iterations. To avoid cycling, since a point already visited may be generated again, a set of points already visited are stored in a list, called Tabu List (TL). The solutions in Y that belong to the TL are eliminated. This TS structure is called short-term memory TS. The use of this type of flexible memory is of great advantage in contrast to the rigid structures of large memory, like those present in Branch-and-Bound methods, or the lack of memory that exists in the Simulated Annealing method [3]. To improve performance, long-term memory TS structures have been proposed to record important attributes like elite and frequently visited solutions. The Directed Tabu Search [16] implemented in this paper for solving nonlinear systems of equations contains long-term memory structures. This is important since the method is able to keep diversity, like population-based methods.

2.2 Directed Tabu Search

The Directed Tabu Search method of Hedar and Fukushima [16] uses direct search methods in order to stabilize the search especially in the vicinity of a local minimum. Two variants of the DTS are therein proposed: one is based on the Nelder-Mead (NM) method, as a local search, inside the exploration step of the algorithm, and the other uses the Adaptive Pattern Search (APS) strategy in the exploration step. Furthermore, the Kelley´s modification of the NM method [19] is still used in the therein called intensification search in the final stage of the process. We note that the DTS method can be classified as a multi-start method. The multi-start methods are designed to build powerful search procedures and guided by a global exploration and local search. These multi-start methods have been successfully applied either in nonlinear global optimization problems or in combinatorial optimization.

The DTS method is based on three procedures: exploration, diversification and intensification search procedures. The structure of the DTS is shown below in Algorithm 1.

Algorithm 1 (DTS method)

Step 1 Randomly generate an initial solution
Step 2 Exploration search procedure
  Step 2.1 Neighborhood search
  Step 2.2 Local search
  Step 2.3 Solution update
  Step 2.4 If *improvement criteria* are reached, repeat from *Step 2.1*
Step 3 Diversification search procedure
Step 4 If *stopping criteria* are not reached, repeat from *Step 2*
Step 5 Intensification search procedure

The main loop (outer cycle) of the DTS method, consisting of the exploration and diversification search procedures, begins with an initial solution. A brief description of the procedures follows. Other mathematical details can be found in [16].

2.2.1 Exploration Search

The exploration search aims to explore the solution space. It uses direct search methods as neighborhood search and local search strategies to generate trial points. These may be based on either the simplex method of NM or on the APS strategy. Here, we use the APS variant. This variant of the DTS mainly focuses its strategy on the definition of an approximate descent direction (ADD), *v*, for the fitness Ψ, as proposed in [15]. Thus, based on a set of *n* trial points, y_i, $i=1,\ldots,n$, using the standard pattern directions, in the neighborhood of the current solution *x*, the descent direction is computed as follows:

$$v = \sum_{i=1}^{n} w_i u_i \quad \text{where} \quad w_i = \frac{\Psi(y_i) - \Psi(x)}{\sum_{j=1}^{n} \left|\Psi(y_i) - \Psi(x)\right|} \quad \text{and} \quad u_i = -\frac{(y_i - x)}{\left\|y_i - x\right\|}, \; i = 1,\ldots,n.$$

Briefly, pattern directions are constructed parallel to the coordinate axes, and a set of points are generated in the neighborhood of the current solution *x*, along these directions with appropriate step length.

Moreover, anti-cycling is prevented not only with the standard Tabu List but also with the inclusion of novel Tabu Regions (TR). The DTS method implements four new TS memory elements.

- The multi-ranked Tabu List (r-TL) is a set of some visited solutions that are ranked and saved according to two features separately, namely: i) ranked in ascending order according to their recency, ii) ranked in ascending order according to their Ψ values.
- Further, two types of regions, around each solution saved in the r-TL are defined:
  - the Tabu Region (TR) in which no new trial point is allowed to be generated,
  - the Semi-Tabu Region (STR), which is a surrounding region around TR, with a radius from its center greater than the radius of the TR, that aims to allow the generation of neighboring trial points when the trial solution lies inside STR.
- Finally, the other memory element is the Visited Region List (VRL) that contains information concerned with the centers of the visited regions and the frequency of visiting these regions. This information is crucial to explore the space outside these visited regions.

These new long-term memory structures are very important since they allow the method in the diversification search and intensification search procedures to behave as an intelligent search technique.

The exploration search procedure is repeatedly applied in order to generate *n* trial solutions using search strategies in the neighborhood of the current solution. If a better move between these raised solutions is found, the current solution is updated and the algorithm continues to the next iteration of this inner cycle (exploration search procedure). Otherwise, the current solution is still better than all the raised solutions in the neighborhood, and the local search strategy generates new local trial points and the current solution is updated with the best trial point generated. The multi-ranked Tabu List is updated and if a new region is reached, the Visited Regions List is updated with information about this region. When the number of iterations in the inner cycle exceeds a pre-specified value or improvement has not been obtained in some consecutive iterations (the *improvement criteria* in *Step 2.4* of Algorithm 1), the diversification search procedure is applied to locate a new starting point, from which the exploration search procedure is repeated.

2.2.2 Diversification Search

A diversification procedure aims to generate a new initial trial point outside the visited regions. The information stored in the VRL is used to direct the search towards new regions. The VRL serves as a diversification tool in the search, with the aim of diversifying the search for areas that have not been visited in the solution space.

2.2.3 Intensification Search

When one of the best obtained trial solutions is sufficiently close to a global minimum, or its value has not been changed for *Nmax* iterations (*stopping criteria* in the *Step 4* of Algorithm 1), then the intensification search procedure is applied at the final stage to refine the best solution visited so far. In this case, a local direct search method, the Kelley´s modification of the Nelder-Mead method [19] and [20], is used. A solution still closer to the global minimum is then obtained.

3 Computational Experiments

We selected, and coded in MATLAB®, 99 test problems from the literature and solved them using *fsolve* from MATLAB, the extended version of the DTS method [16], and a SA version for comparative purposes. The problems in our database are referred to as P1, P2, …, P99. They represent systems of nonlinear equations of different sizes and complexity. The results of the numerical experiments were obtained in a personal computer with an AMD Turion 2.20 GHz processor and 3 GB of memory. Due to the stochastic nature of DTS and SA algorithms, each problem was run 30 times and the best of the 30 solutions, as well as the average of the 30 obtained solutions were registered. The DTS algorithm starts with an initial solution, which is randomly generated inside the range [*l*,*u*]. Although not all problems defined in the literature have a registered

[*l*,*u*], we selected a specific range for each problem, as shown in Table 4 in the Appendix. The lower and upper limits are the same for all the components of the solution. The parameters defined for the *stopping criteria* of the algorithm are $Nmax = 10\,n$ and $\varepsilon = 10^{-8}$ (in *Step 4* of Algorithm 1), where the condition that defines the closeness of the best solution to the global minimum is the following

$$\Psi(x_k) \leq \varepsilon .$$

The goal of this paper is twofold:

i) to evaluate the effectiveness of the extended DTS to solve systems of nonlinear equations that are not solved by Newton´s method;

ii) to analyze the effect of the dimension of the problem in the performance of the extended DTS algorithm.

Table 2. Comparison of *fsolve*, DTS and SA

		fsolve			DTS			SA		
	n	$\Psi(x)$	*nfe*	*nit*	Ψ_{avg}	*a-nfe*	Ψ_{min}	Ψ_{avg}	*a-nfe*	Ψ_{min}
P6	2	**0.0025**	57	20	0.5747	327	0.019	0.6677	265	0.0715
P9	2	6.9989	49	20	3.2664	242	1.8e-04	4.1994	243	**2.9e-05**
P25	10	13.830	1001	90	0.1227	5101	0.0142	0.4112	1014	**0.0049**
P26	3	0.6642	106	30	0.4195	482	**9.5e-06**	0.2428	303	1.3e-05
P27	3	0.1008	109	30	0.0563	667	**0.0011**	0.0669	339	0.0015
P44	33	2.1965	1170	47	1.5666	75246	**0.6737**	3.5248	3302	0.7079
P45	33	**0.3983**	3306	104	5.7368	48664	3.5176	6.1427	3302	4.2613
P50	2	**0.2476**	47	20	0.24	247	**0.2476**	0.2476	240	**0.2476**
P52	5	0.0022	261	50	4.0e-02	1556	5.0e-04	8.2e-02	507	**2.0e-04**
P53	6	2.8074	427	60	0.0015	1958	**0.0001**	0.0019	604	**0.0001**
P55	6	138.42	385	60	4.6e+06	3634	**122.84**	8.0e+06	631	129.65
P56	8	0.4919	609	80	1.1e+01	4355	**0.4286**	1.1439	802	0.4308
P62	6	**0.0467**	379	60	5.4e+10	4512	45.353	5.1e+11	616	44.31
P63	6	0.0062	427	60	0.0020	2030	**5.4e-05**	0.0177	602	0.0002
P64	8	3.2592	657	80	0.4907	3216	**5.1e-05**	3.1581	887	0.0003
P70	10	25.782	981	100	53.5830	5870	**9.0034**	47.1696	1002	**9.0034**
P72	51	2.7307	1989	50	1.9574	151108	**1.1470**	2.0317	5102	1.4808
P77	2	4.5632	45	20	0.8533	243	**0.7377**	1.0068	211	**0.7377**
P80	3	0.0630	124	30	2.0e-05	516	5.2e-06	0.1139	325	**3.9e-06**
P84	3	0.0005	124	30	3.1e-08	482	1.4e-09	5.0e-08	144	**1.2e-09**
P88	10	0.7460	66	5	8.6e-03	4773	1.4e-03	8.8e-03	1002	**7.7e-04**
P96	2	2.8098	47	20	0.1040	258	**1.7e-05**	0.4617	265	1.8e-05
P99	2				0.0600	252	**4.0e-06**	9.6e-02	292	1.1e-05

To resume the main achievements of our numerical experiments, we show in Table 2 the results of the 23 problems (from the database of 99 problems) that

were selected because *fsolve* was not able to converge to the required solution with a tolerance of $\varepsilon = 10^{-8}$ and within $10\,n$ iterations ("exitflag" is "-2" for P44 and P72 and is "0" for all the others). Here, we use the initial approximation provided in the literature. Problem P99 of our list is the Example 1 described in Section 1.

The characteristics of the selected problems are listed in the Appendix of the paper (Table 4). Moreover, the results of a selection of other problems from our database with varied dimensions are shown in Table 3.

Table 3. Comparison based on problems with varied dimensions

		fsolve			DTS			SA		
	n	$\Psi(x)$	*nfe*	*nit*	Ψ_{avg}	*a-nfe*	Ψ_{min}	Ψ_{avg}	*a-nfe*	Ψ_{min}
P14	2	**2.8e-08**	19	6	0.0092	251	4.4e-06	4.6e-05	287	2.3e-06
	10	**3.7e-08**	55	4	0.93	4918	0.0023	0.8545	1022	0.0031
	30	**8.6e-14**	155	4	5.8	50073	1.9910	5.2442	3002	1.3550
P15	2	**0**	15	4	4.9e-05	246	5.6e-07	3.5e-05	258	1.6e-06
	10	**7.2e-07**	101	10	0.7422	4731	4.7e-05	20.1567	1040	2.7e-05
	30	**4.1e-08**	311	10	2.220	39478	0.0023	1.0e+07	3257	26.4957
P16	2	**3.5e-07**	9	2	4.1e-05	250	8.8e-06	3.2e-05	232	3.8e-06
	10	**3.1e-08**	33	2	0.006	4915	0.0006	0.005	1002	0.0009
	30	**5.1e-05**	62	1	0.25	44379	0.1201	0.2868	3002	0.0934
P17	2	**3.5e-07**	9	2	3.0e-05	249	3.7e-06	1.6e-05	246	1.9e-06
	10	**3.7e-07**	33	2	4.8e-04	3978	8.3e-05	0.0003	1002	7.3e-05
	30	**6e-07**	93	2	0.015	36010	0.0012	0.0045	3002	0.0009
P18	2	**5e-09**	15	4	2.6e-02	252	1.4e-06	3.8e-05	223	1.1e-06
	10	**1.1e-09**	55	4	1.00	5067	0.6301	1.0404	1002	0.9375
	30	**4.2e-14**	186	5	1.2	40639	1.0569	1.322	3002	1.0568
P19	2	**2.6e-08**	18	5	4.3e-05	259	6.8e-06	7.9e-05	212	1.0e-05
	10	**1.5e-08**	66	5	1.6	5108	1.7e-05	1.6027	1002	0.0073
	30	**5e-12**	217	6	1.9	50845	0.0026	1.753	3002	1.753
P21	4	**8.1e-15**	41	8	4.4e-04	962	1.7e-04	6.7e-04	426	6.8e-05
	52	**1.7e-14**	424	7	2.8884	156373	0.5985	6.8724	5202	3.4877
P22	2	**0.0001**	6	1	8.4128	296	0.0722	18023.3	314	0.0793
	50	**0.0005**	102	1	6.5e+07	98182	1.1e+04	3.3e+05	5002	7.5e+03
P26	3	**0.6642**	106	30	0.4195	482	9.5e-06	0.2428	303	1.3e-05
	33	**2.1965**	1170	47	1.5666	75146	0.6737	3.5248	3302	0.7079

Tables 2 and 3 summarize the numerical results, where *n* represents the dimension (number of variables = number of functions in the system), $\Psi(x)$ is the value of the l_2-norm error function at the found solution, *nfe* is the number of function evaluations and *nit* the number of iterations, all provided by MATLAB; Ψ_{avg} represents the average of the obtained solutions over the 30 runs, *a-nfe*

gives the average number of function evaluations computed over the 30 runs (rounded to the nearest integer) and Ψ_{min} is the best solution obtained during the 30 runs. The best solutions found in these comparisons are printed in "bold". We compare $\Psi(x)$ of *fsolve* with Ψ_{min} of DTS and SA.

4 Conclusions

In this paper we show that nonlinear systems of equations can be effectively solved by implementing a global optimization method to a fitness function, which represents the l_2-norm error function related to the solving of the system of equations. The application of an extended version of the metaheuristic Directed Tabu Search, proposed in [16], for solving complex and difficult nonlinear systems of equations has been analyzed and tested. From Table 2, we may conclude that DTS is mostly able to converge to the solutions of the selected systems that are not solved by a Newton-type method. However, the results of Table 3 are not so promising. As expected, when a Newton-type method converges, the accuracy of the found solution is higher than that obtained by a metaheuristic. Furthermore, the performance of both tested metaheuristics (DTS and SA) is greatly affected by the dimension of the problem. This suggest that a combination of global- and local-type search procedures, carried out in separate iterations, depending on the need for an exploration of the search space or an exploitation of a promising region in the search space, will improve performance. For the local-type iteration, a derivative local search method seems crucial, as long as analytical or numerical derivatives could be used. This issue will be addressed in the near future.

Appendix – Problems used in numerical experiments

Table 4. Characteristics of the problems

	n	[*l*,*u*]	reference	Problem name in cited paper
P6	2	[-10,10]	[27]	P3-*Powell badly scaled function*
P9	2	[-10,15]	[27]	P2-*Freudenstein and Roth function*
P14	2,10,30	[-10,10]	[27]	P26-*Trignometric function*
P15	2,10,30	[-10,10]	[27]	P27-*Brown almost-linear function*
P16	2,10,30	[-10,10]	[27]	P28-*Discrete boundary value function*
P17	2,10,30	[-10,10]	[27]	P29-*Discrete integral equation function*
P18	2,10,30	[-10,10]	[27]	P30-*Broyden tridiagonal function*
P19	2,10,30	[-10,10]	[27]	P31-*Broyden banded function*
P21	4,52	[-5,5]	[8]	D2-*Augmented Rosenbrock*
P22	2,50	[0,370]	[8]	D3-*Powell badly scaled*
P25	10	[10,50]	[8]	D6-*Shifted and augmented trigonometric function with an Euclidian sphere*
P26/P44	3/33	[-10,10]	[8]	D7-*Diagonal of three variables premultiplied by a quasi-orthogonal matrix*
P27/P45	3/33	[-10,10]	[8]	D8-*Diagonal of three variables premultiplied by an orthogonal matrix, combined with inverse trigonometric function*
P50	2	[-10,10]	[18]	pp. 2004
P52	5	[-20,20]	[26]	*Combustion of propane chemical equilibrium equations*
P53	6	[-3,3]	[27]	14-*Wood function*
P55	6	[-10,10]	[29]	*Semicoductor boundary condition*
P56	8	[-10,30]	[1]	2.3-*The human heart dipole*

Table 4. Characteristics of the problems (cont.)

	n	[*l*,*u*]	reference	Problem name in cited paper
P62	6	[0,60]	[17]	*Problem 2*
P63	6	[-10,10]	[23]	*Example 2*
P64	8	[-10,15]	[7]	*Equation 3.1*
P70	10	[-100,100]	[30]	*Example 4.1-Nonlinear resistive circuit*
P72	51	[-5,5]	[8]	D7-*Diagonal of three variables premultiplied by a quasi-orthogonal matrix*
P77	2	[0,3.5]	[4]	*Example 1*
P80	3	[0,4]	[27]	5-*Beale function*
P84	3	[0,1]	[22]	*Example 6.2*
P88	10	[-10,10]	[21]	*Example 2-The Beam problem*
P96	2	[-10,10]	[12]	pp. 498 (*Problem N4*)

Acknowledgement The first author is supported by the CIDEM - Centre for Research & Developement in Mechanical Engineering, from Portugal.

References

[1] B.M. AVERICK, R.G. CARTER AND J.J. MORÉ, *The MINPACK-2 test problem collection* (*Preliminary version*), Technical Memorandum n. 150, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, 1991.

[2] E. BODON, A. DEL POPOLO, L. LUKŠAN AND E. SPEDICATO, *Numerical performance of ABS codes for systems of nonlinear equations*, Technical Report DMSIA 01/2001, Universitá degli Studi di Bergamo, Bergamo, Italy, 2001.

[3] L. CAVIQUE, C. REGO AND I. THEMIDO, *Estruturas de vizinhança e procura local no problema da clique máxima*, Investigação Operacional, **22**, (2002) 1-18.

[4] C.H. CHEN, *Finding roots by genetic algorithms*, In: 2003 Joint Conference on AI, Fuzzy System and Gray System, Taipei, Taiwan, (2003) 4-6.

[5] D. CVIJOVIĆ AND J. KLINOWSKI, *Taboo search: an approach to the multiple minima problem*, Science, **267**, (1995) 664-666.

[6] J.E. DENNIS AND R.B. SCHNABEL, *Numerical methods for unconstrained optimization and nonlinear equations*, Prentice-Hall, Inc., 1983.

[7] M. DRĂGAN, *On solving large sparse systems of nonlinear equationsusing threads*, In: Proceedings of NMCM2002, An Euro Conference on Numerical Methods and Computational Mechanics, Miskolc, Hungary, (2002) 49-56.

[8] A. FRIEDLANDER, M.A. GOMES-RUGGIERO, D.N. KOZAKEVICH, J.M. MARTINEZ AND S.A. SANTOS, *Solving nonlinear systems of equations by means of Quasi-Newton methods with a nonmonotone strategy*, Optimization Methods and Software, **8**, (1997) 25-51.

[9] F. GLOVER, *Future paths for integer programming and links to artificial intelligence*, Computers and Operations Research **13**, n. 5, (1986) 533-549.

[10] F. GLOVER AND M. LAGUNA, *Tabu Search*, Kluver Academic Publishers, 1997.

[11] D.E. GOLDBERG, *Genetic Algorithms in Search, Optimization \& Machine Learning*, Addison-Wesley (1989).

[12] M.D. GONZÁLEZ-LIMA AND F.M. OCA, *A Newton-like method for nonlinear system of equations*, Numer. Algor. **52** (2009) 479-506.

[13] C. GROSAN AND A. ABRAHAM, *A new aproach for solving nonlinear equations systems*, IEEE Transactions on systems, man, and cybernetics – part A: systems and humans, **38**, n. 3, (2008) 698-714.

[14] C. GROSAN AND A. ABRAHAM, *Multiple solutions for a system of nonlinear equations*, International Journal of Innovative Computing, Information and Control, (2008).
[15] A. HEDAR AND M. FUKUSHIMA, *Heuristic Pattern Search and its Hybridization with Simulated Annealing for Nonlinear Global Optimization*, Optimization Methods and Software, **19**, (2004) 291-308.
[16] A. HEDAR AND M. FUKUSHIMA, *Tabu search direct by direct search methods for nonlinear global optimization*, European Journal of Operational Research, **170**, (2006) 329-349.
[17] K.L. HIEBERT, *An evaluation of mathematical software that solves systems of nonlinear equations*, ACM Transactions on Mathematical Software, **8**, n.1, (1982) 5-20.
[18] M.L. HIRSCH, P.M. PARDALOS AND M. RESENDE, *Solving systems of nonlinear equations with continuous GRASP*, Nonlinear Analysis: Real World Applications, **10,** (2009) 2000-2006.
[19] C.T. KELLEY, *Detection and remediation of stagnation in the Nelder-Mead algorithm using a sufficient decrease condition*, SIAM Journal on Optimization, **10**, (1999) 43-55.
[20] C.T. KELLEY, *Iterative Methods for Optimization*, Frontiers Appl. Math., Vol. 18, SIAM, Philadelphia, PA, 1999.
[21] C.T. KELLEY, L. QI, X. TONG AND H. YIN, *Finding a stable solution of a system of nonlinear equations arising from dynamic systems,* Journal of Industrial and Management Optimization, **7**, n. 2, (2011) 497-521.
[22] L.N. KOLEV, *Interval methods for circuit analysis*, World Scientific, Singapore, 1993.
[23] L.V. KOLEV, *An improved interval linearization for solving nonlinear problems*, Numerical Algorithms, **13**, n. 1-4, (2004) 213-224.
[24] J.M. MARTINEZ, *Algorithms for solving nonlinear systems of equations,* In Continuous Optimization: the state of art, edited by E. Spedicato. Kluwer Academic Publishers, (1994) 81-108.
[25] J.M. MARTINEZ, *Practical quasi-Newton methods for solving nonlinear systems*, Journal of Computational and Applied Mathematics, **124**, (2000) 97-122.
[26] K. MEINTJES AND A.P. MORGAN, *Chemical equilibrium systems as numerical test problems*, ACM Transaction on Mathematical Software, **16**, n. 2 (1990) 143-151.
[27] J.J. MORÉ, B.S. GARBOW AND K.E. HILLSTROM, *Testing unconstrained optimization software*, ACM Transactions on Mathematical Software, **7**, n. 1, (1981) 17-41.
[28] J.A. NELDER AND R. MEAD, *A simplex method for function minimization*, Computing Journal, **7**, (1965) 308-313.
[29] U. NOWAK AND L. WEIMANN, *A family of Newton codes for systems of highly nonlinear equations*, Technical Report. TR-91-10, 1991.
[30] K. YAMAMURA, H. KAWATA AND A. TOKUE, *Interval solution of nonlinear equations using linear programming*, BIT – Numerical Mathematics, **38**, n. 1, (1998) 186-199.