



Process Systems Engineering and Process Safety

Nonlinear constrained optimization using the flexible tolerance method hybridized with different unconstrained methods

Alice Medeiros Lima^{1,*}, Antonio José Gonçalves Cruz^{1,2}, Wu Hong Kwong²¹ Chemical Engineering Graduate Program, Federal University of São Carlos, Rodovia Washington Luiz, km 235 — SP 310, São Carlos, SP 13565-905, Brazil² Department of Chemical Engineering, Federal University of São Carlos, Rodovia Washington Luiz, km 235 — SP 310, São Carlos, SP 13565-905, Brazil

ARTICLE INFO

Article history:

Received 28 April 2016

Received in revised form 26 July 2016

Accepted 31 July 2016

Available online 4 September 2016

Keywords:

Flexible tolerance method

Modified Powell's method

BFGS

PSO

Scaling

Hybridization

ABSTRACT

This paper proposes the use of the flexible tolerance method (FTM) modified with scaling of variables and hybridized with different unconstrained optimization methods to solve real constrained optimization problems. The benchmark problems used to analyze the performance of the methods were taken from *G-Suite* functions. The original method (FTM) and other four proposed methods: (i) FTM with scaling of variables (FTMS), (ii) FTMS hybridized with BFGS (FTMS-BFGS), (iii) FTMS hybridized with modified Powell's method (FTMS-Powell) and (iv) FTMS hybridized with PSO (FTMS-PSO), were implemented. The success rates of the methods were 80%, 100%, 75%, 95% and 85%, for FTM, FTMS, FTMS-BFGS, FTMS-Powell and FTMS-PSO, respectively. Numerical experiments including real constrained problems indicated that FTMS gave the best performance, followed by FTMS-Powell and FTMS-PSO. Despite the inferior performance compared to FTMS and FTMS-Powell, the FTMS-PSO method presented some advantages since good different initial points could be obtained, which allow exploring different routes through the solution space and to escape from local optima. The proposed methods proved to be an effective way of improving the performance of the original FTM.

© 2016 The Chemical Industry and Engineering Society of China, and Chemical Industry Press. All rights reserved.

1. Introduction

Constrained nonlinear optimization problems can be solved using a wide range of optimization methods. These can be broadly divided into deterministic and stochastic methods. In deterministic methods, every step can be predicted from knowledge of the starting point, so that, the same answer is always obtained when beginning from the same starting point. However, in stochastic methods, several choices are made based on random numbers drawn at the time of code execution. Since, at every code execution, all the numbers will be different, a random method does not perform the same sequence of operations in two successive runs. Starting from the same point, each code execution will follow its own path and possibly lead to a different final answer.

The search for global optimality in nonlinear optimization problems is a challenge for any optimization method. For linear problems, finding the global optimum can, in principle, be guaranteed. However, in nonlinear problems there is no guarantee of finding the global optimum, even using deterministic and/or stochastic methods, [1].

The flexible tolerance method (FTM), which was the main method used in this work, is a deterministic method that does not use derivative information to perform the search. Although the FTM was first proposed

in the 1960s by Paviani and Himmelblau [2], the method has been little explored and there are few applications reported in the literature. However, the use of decreasing infeasibility tolerances has been employed in many modern methods for constrained optimization, as reported by Martínez and Sobral [3].

Fenton *et al.* [4] proposed a modification in FTM and apply the method in the solution of NLP optimization problems. In the modification, the random search technique is used to complement the flexible tolerance method by generating a datafile which contains feasible solutions of the optimization problem in question. According to the authors, the modification proposed improved the efficiency and effectiveness of the flexible tolerance method.

Zhang and Ren [5] applied FTM to multicomponent spectrophotometric analyses. It was found that FTM was an efficient procedure for optimization of multidimensional problems, especially in the presence of interactions between variables and when the problem was ill-conditioned.

Constantinescu [6] used FTM to determine smooth and time-optimal path-constrained trajectories for robotic manipulators. The method was chosen for two reasons: the derivatives of the constraints and the cost function were not available; and the solution sought was expected to be on the boundary of the admissible region (in which case it is desirable to use information about points on both sides of the limiting surface in order to converge to the surface). The FTM proved to be efficient in solving the problem of trajectories with time optimality and smoothness.

* Corresponding author.

E-mail address: alice.medeirosdelima@gmail.com (A.M. Lima).

Chen and Yin [7] used the FTM to solve the optimization problems for determining hydrologic parameters in the root zone: water uptake rate, spatial root distribution, infiltration rate, and evaporation.

Shang *et al.* [8] used FTM with an AGA (adaptive genetic algorithm) to solve nonlinear, multimodal and multi-constraint optimization problems. FTM, serving as one of the AGA operators, used a flexible tolerance criterion for near-feasible points to minimize the constraint violation of an objective function. Complex functions were evaluated, and the authors concluded that the hybrid method was suitable for resolution of real-world problems.

Omowunmi and Susu [9] used FTM to estimate the kinetic parameters of an autocatalytic reaction involving the pyrolysis of n-eicosane. The method effectively minimized the sum of squares residuals between the experimental and predicted rates of reaction.

In previous work, Lima *et al.* [10] used FTM in the synthesis and analysis of processes with mass integration. The FTM was compared with two indirect optimization methods (GRG and SQP) and good results were obtained solving a classic case of mass integration. It was found that when the FTM was applied to solve engineering problems, the inner search became slow and difficult when the dimensions of the

problem became larger. Another difficulty concerned variations involving the range of variables, which could make it hard for the FTM to achieve convergence. In order to try to resolve these issues, the present work proposes the transformation of variables by scaling, as well as hybridization with other methods. Selection of the methods used for hybridization with FTM was based on: (i) derivative-free use, (ii) ease of implementation and (iii) good performance reported in the literature. The methods proposed in this work was based on flexible tolerance method (FTM), and were (i) FTM with scaling of variables (FTMS), (ii) FTMS hybridized with BFGS (FTMS-BFGS), (iii) FTMS hybridized with modified Powell's method (FTMS-Powell) and (iv) FTMS hybridized with PSO (FTMS-PSO).

2. Methodology

The general formulation of a nonlinear programming problem can be stated by Eqs. (1)–(3) as follows:

$$\text{Minimize } f(\mathbf{x}) \quad \mathbf{x} \in \mathcal{R}^n \quad (1)$$

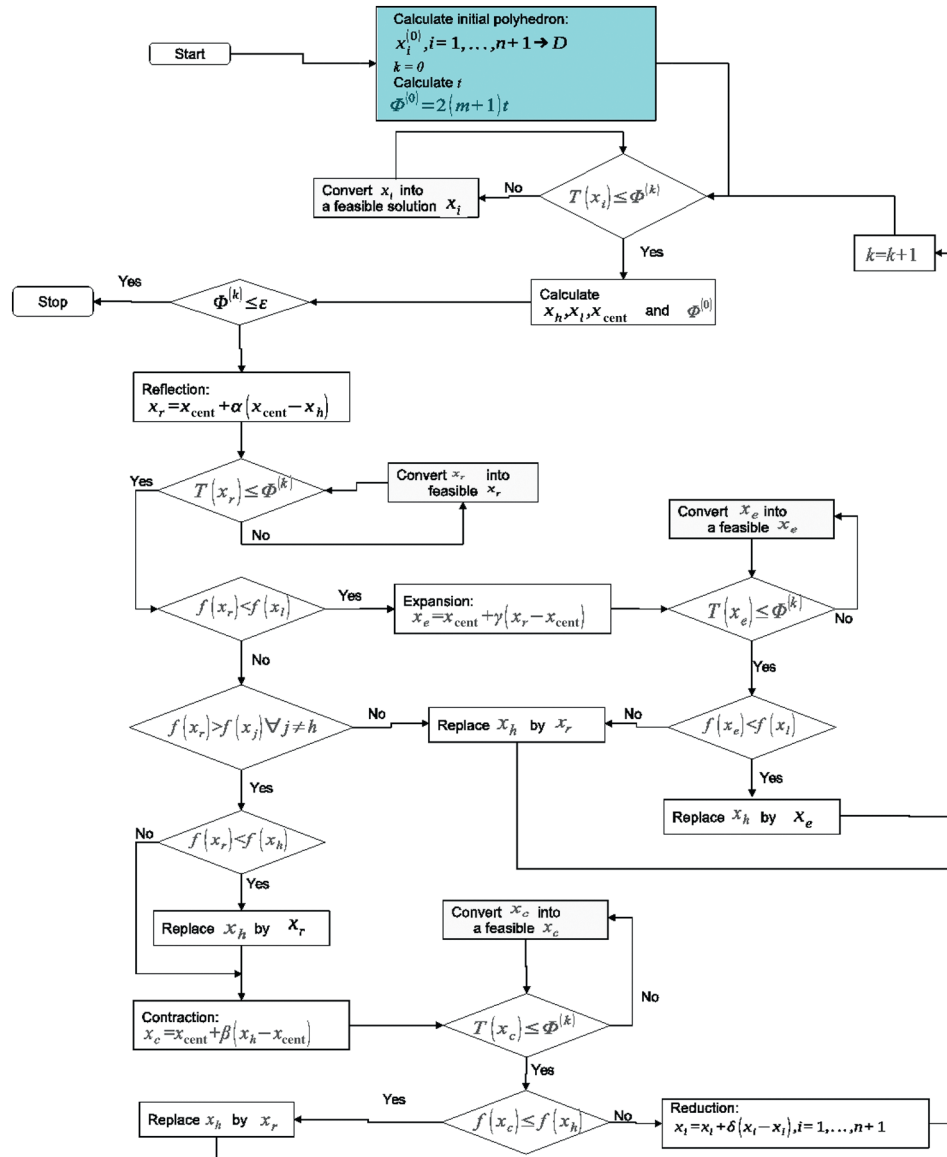


Fig. 1. FTM algorithm flowchart for performing the outer search that minimizes the objective function $f(\mathbf{x})$. All vectors $\mathbf{x}$ are assumed to represent $\mathbf{x}^{(k)}$, unless noted otherwise. Adapted and modified from Naish [11]. The gray boxes indicated the inner search that can be performed using different unconstrained minimization algorithms (in this work: Nelder–Mead, Powell and BFGS). The blue box indicates the initialization step that is replaced by PSO method in this work.

$$\text{Subject to: } h_i(\mathbf{x}) = 0 \quad i = 1, \dots, m \quad (2)$$

$$g_i(\mathbf{x}) \geq 0 \quad i = m + 1, \dots, p \quad (3)$$

where $f(\mathbf{x})$ is the objective function, $h_i(\mathbf{x})$ the equality constraints and $g_i(\mathbf{x})$ the inequality constraints.

The flexible tolerance method (FTM) is a direct search method and was proposed by Paviani and Himmelblau [2]. The FTM algorithm improves the value of objective function using the information provided by feasible points, as well some non-feasible points, called near-feasible points. The feasibility becomes more restrictive as the search moves towards the problem solution, until the limit where only the feasible vectors $\mathbf{x}$ of Eqs. (1)–(3) are accepted. The result of this basic strategy, the optimization problem, Eqs. (1)–(3), can be rewritten as indicated in Eqs. (4) and (5):

$$\text{Minimize: } f(\mathbf{x}) \quad \mathbf{x} \in \mathcal{R}^n \quad (4)$$

$$\text{Subject to: } \Phi^{(k)} - T(\mathbf{x}) \geq 0 \quad (5)$$

where $\Phi^{(k)}$ is the flexible tolerance criterion for feasibility in the k stage of the search, and $T(\mathbf{x})$ is the positive functional of all constraints of equality and/or inequality of the problem, used as a measure of the violation extent of constraints. The functional $T(\mathbf{x})$ is described by Eq. (6), where U is the Heaviside operator according to Eq. (7).

$$T(\mathbf{x}) = \left[\sum_{i=1}^m h_i^2(\mathbf{x}) + \sum_{i=m+1}^p U g_i^2(\mathbf{x}) \right]^{1/2} \quad (6)$$

$$U_i = \begin{cases} 0 & \text{for } g_i(\mathbf{x}) \geq 0, \\ 1 & \text{for } g_i(\mathbf{x}) < 0 \end{cases} \quad (7)$$

The tolerance criterion at the k th iteration is given by Eq. (8), where m is the number of equality constraints, $r = (n - m)$ the number of degrees of freedom of $f(\mathbf{x})$, $\mathbf{x}_i^{(k)}$ the i th vertex of the polyhedron in $\mathcal{R}^n$, $\mathbf{x}_{\text{cent}}^{(k)}$ the vertex corresponding to the centroid, t the size of the initial polyhedron and k the interaction index.

$$\Phi^{(k)} = \min \left\{ \Phi^{(k-1)}, \frac{m+1}{r+1} \sum_{i=1}^{r+1} \left\| \mathbf{x}_i^{(k)} - \mathbf{x}_{\text{cent}}^{(k)} \right\| \right\} \quad (8)$$

$$\Phi^{(0)} = 2(m+1)t$$

The tolerance criterion $\Phi^{(k)}$ behaves as a positive decreasing function of $\mathbf{x}$, and the vector $\mathbf{x}^{(k)}$ is classified as follows:

- Feasible, if $T(\mathbf{x}^{(k)}) = 0$.
- Near-feasible, if $0 < T(\mathbf{x}^{(k)}) \leq \Phi^{(k)}$.
- Non-feasible, if $T(\mathbf{x}^{(k)}) > \Phi^{(k)}$.

The tolerance for near-feasible solutions is decreased until the limit when only feasible solutions are allowed, according to Eq. (9), where $\mathbf{x}^*$ is the solution vector, within tolerance ε .

$$\lim_{\mathbf{x} \rightarrow \mathbf{x}^*} \Phi^{(k)} = 0 \quad (9)$$

The FTM performs an outer search that minimizes the objective function $f(\mathbf{x})$ and an inner search that minimizes the value of $T(\mathbf{x})$. The outer search (Fig. 1) is a variation of the Nelder–Mead method, when a new vertex is founded during the search, its feasibility is evaluated. If the vertex is near feasibility the search continues. If the outer search select a non-feasible vertex, an inner search (indicated by gray boxes in Fig. 1) is performed to convert this non-feasible vertex in a near-feasible or feasible vertex.

The inner search that minimizes the value of $T(\mathbf{x})$ can be performed using any multi-variable search technique. In the original implementation [2] the inner search uses the Nelder–Mead method (or Flexible Polyhedron Method, FPM), Algorithm 1. The Nelder–Mead method minimizes the function $T(\mathbf{x})$ of n independent variables using $n + 1$ vertices of a flexible polyhedron in $\mathcal{R}^n$. This standard formulation is designated as FTM.

In addition to the standard FTM method and the FTM method with scaling of variables (FTMS), additional variations used in this work employed other methods to perform the inner search. These methods were FTMS hybridized with the BFGS method (FTMS-BFGS), and FTMS hybridized with the Powell method (FTMS-Powell). FTMS was also hybridized with the PSO method (FTMS-PSO), where the PSO method was used to perform the initialization. These methods are described in the next section.

Algorithm 1. Pseudocode of Nelder–Mead method.

Start: Calculate initial simplex: $\mathbf{x}_j^0, j = 1, \dots, n + 1 \rightarrow D$. Set $k = 0$

Step 1: Calculate $\mathbf{x}_h, \mathbf{x}_l$ and $\mathbf{x}_{\text{cent}}$

Step 2: Reflection: $\mathbf{x}_r = \mathbf{x}_{\text{cent}} + \alpha(\mathbf{x}_{\text{cent}} - \mathbf{x}_h)$

if $T(\mathbf{x}_r) < T(\mathbf{x}_l)$ **then**

Step 3: Expansion: $\mathbf{x}_e = \mathbf{x}_{\text{cent}} + \gamma(\mathbf{x}_r - \mathbf{x}_{\text{cent}})$

if $T(\mathbf{x}_e) > T(\mathbf{x}_l)$ **then**

Step 6: Replace $\mathbf{x}_h$ by $\mathbf{x}_e$

else

Step 7: Replace $\mathbf{x}_h$ by $\mathbf{x}_r$

end if

else

if $T(\mathbf{x}_r) > T(\mathbf{x}_j) \forall j \neq h$ **then**

if $T(\mathbf{x}_r) < T(\mathbf{x}_h)$ **then**

Step 4: Replace $\mathbf{x}_h$ by $\mathbf{x}_r$

end if

Step 5: Contraction: $\mathbf{x}_c = \mathbf{x}_{\text{cent}} + \beta(\mathbf{x}_h - \mathbf{x}_{\text{cent}})$

else

Go to Step 7.

end if

if $T(\mathbf{x}_c) \leq T(\mathbf{x}_h)$ **then**

Step 8: Replace $\mathbf{x}_h$ by $\mathbf{x}_c$

else

Step 9: Reduction: $\mathbf{x}_j = \mathbf{x}_l + \delta(\mathbf{x}_j - \mathbf{x}_l), j = 1, 2, \dots, n + 1$

end if

if $\sqrt{\frac{1}{n+1} \sum_{j=1}^{n+1} [T(\mathbf{x}_j) - T(\mathbf{x}_{\text{cent}})]^2} \leq \varepsilon$ **then**

Stop.

else

$k = k + 1$. **Go to Step 1.**

end if

end if

The codes used in this work were implemented in Python™ and the optimization calculations were performed using Eclipse® IDE software, run under a Linux-like operating system installed on a 2 GHz Pentium® Dual-Core computer. The FTM parameters used were $\alpha = 1.0$, $\beta = 0.5$, $\gamma = 2.0$ and $\delta = 0.5$ as recommended previously, [12]. The tolerance adopted was $\varepsilon = 10^{-5}$, and the size of the initial polyhedron was $t = 0.4$.

The benchmark chosen for analysis of the performance of the optimization algorithms was the net of functions presented in CEC'06 [13], which cover a wide range of nonlinear constrained optimization problems, with real problems and some generated problems, and different

constraint types (physical, time, geometric, etc.). Two problems of chemical engineering interest, namely heat exchanger design and chemical equilibrium, were detailed in the results.

2.1. Proposed methods

2.1.1. FTMS

The standard FTM using variable scaled is designated FTMS. The scaled variable (y_j) is defined according Eq. (10), as reported by Gill *et al.* [14], where U_j and L_j are the upper and lower bounds of variable (x_j). This transformation guarantees that $-1.0 \leq y_j \leq +1.0$, $\forall j$, regardless of value x_j within the interval $[U_j, L_j]$.

$$y_j = \frac{2x_j}{U_j - L_j} - \frac{L_j + U_j}{U_j - L_j} \quad (10)$$

2.1.2. FTMS-BFGS

The BFGS (Broyden, Fletcher, Goldfarb and Shanno) method employs the quasi-Newton procedure, reported by Nocedal and Wright [15]. This method can use the gradient of the objective function. However, since the aim in this work was to use a derivative-free method, estimation was performed using first-differences. This method employed the Scipy¹ library [16], especially the *scipy.optimize* package which provides several commonly used optimization algorithms.

The BFGS method with Scipy library was implemented based on Nocedal and Wright [15], as shown in Algorithm 2. The jacobian of the objective function of the FTMS inner search ($T(\mathbf{x})$) was calculated using forward finite differences, as indicated in Eq. (11), where τ is a small positive scalar, and $\mathbf{e}_i$ is the i th unit vector, for $i = 1, \dots, n$.

$$\nabla T_k = \frac{\partial T}{\partial x_i} \mathbf{x} \approx \frac{T(\mathbf{x} + \tau \mathbf{e}_i) - T(\mathbf{x})}{\tau} \quad (11)$$

Algorithm 2. Pseudocode of Broyden, Fletcher, Goldfarb and Shanno (BFGS) method.

Given starting point $\mathbf{x}_0$, convergence tolerance $\varepsilon > 0$, inverse Hessian approximation H_0

$k = 0$

while $\|\nabla T_k\| > \varepsilon$ **do**

 Compute search direction: $\mathbf{p}_k = -H_k \nabla T_k$

 Set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \cdot \mathbf{p}_k$, where α_k is computed from a line search procedure to satisfy the Wolfe conditions

 Define $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ and $\mathbf{y}_k = \nabla T_{k+1} - \nabla T_k$

 Compute $H_{k+1} = (I - \rho_k \cdot \mathbf{s}_k \cdot \mathbf{y}_k^T) H_k (I - \rho_k \cdot \mathbf{y}_k \cdot \mathbf{s}_k^T) + \rho_k \cdot \mathbf{s}_k \cdot \mathbf{y}_k^T$, where

$$\rho_k = \frac{1}{\mathbf{s}_k \cdot \mathbf{y}_k^T}$$

$k = k + 1$

end while

The BFGS method was employed each time the FTMS called for minimization of the inner search ($T(\mathbf{x})$), indicated by the gray boxes in the algorithm flowchart of Fig. 1.

2.1.3. FTMS-Powell

The method employed was a modification of Powell's method [17], which is a conjugate direction method that, is included in the *scipy.optimize* package from SciPy. It performs sequential one-dimensional minimizations along each vector of the direction set, which is updated at each iteration of the main minimization loop. The

function need not be differentiable, and no derivatives are taken. The pseudocode is shown in Algorithm 3.

Powell's method was employed each time the FTMS called for minimization of the inner search ($T(\mathbf{x})$), indicated by the gray boxes in the algorithm flowchart of Fig. 1.

2.1.4. FTMS-PSO

The PSO was hybridized with the FTMS method in a different way. In codes described previously (FTMS-BFGS and FTMS-Powell), the unconstrained optimization methods were used to perform the inner search. In the case of FTMS-PSO, the PSO method was used during the initialization of the FTMS, in order to find a feasible starting point. The PSO code in its standard form is available in the DEAP (Distributed Evolutionary Algorithms in Python) library that includes several evolutionary optimization algorithms in Python, [18]. The PSO from the DEAP library was the basis for building the PSO algorithm used in this work.

The PSO method employed here used the inertia weight (ω) calculated as indicated in Eq. (12), [19]. After some initial tests, the random term used for the inertia weight calculus was set between 0.1 and 1.2.

$$\omega = 0.5 + \frac{\text{Rnd}(0.1, 1.2)}{2} \quad (12)$$

Algorithm 3. Pseudocode of Powell's modified method.

Given $s_k, k = 1, 2, \dots, n \in \mathbf{P} \subseteq \mathbb{R}^n$, a linearly independent set of vectors in $\mathbf{P}$

$\mathbf{p}_0$ is the starting point

Set $\mathbf{p}_0 = \mathbf{x}_i, i = 0$

for $k = 1, 2, \dots, n$ **do**

 Determine λ_k by minimizing $f(\mathbf{x}_{k-1} + \lambda_k \cdot \mathbf{s}_k)$ defined by $\mathbf{x}_k = \mathbf{x}_{k-1} + \lambda_k \cdot \mathbf{s}_k$

end for

for $k = 1, 2, \dots, n$ **do**

$$\Delta f_k = f(\mathbf{x}_k) - f(\mathbf{x}_{k-1})$$

 Find index r , such that $\Delta f = |\Delta f_r| = \max |\Delta f_r|$ is the magnitude of maximum decrease of f

$\mathbf{s}_r$ is the direction of the maximum decrease about directions of the previous step

end for

repeat

 Set $i = i + 1$

for $k = 1, 2, \dots, n$ **do**

$$f_k = f(\mathbf{x}_k) - f(\mathbf{x}_{k-1})$$

end for

 Set $f_3 = f(2\mathbf{x}_n - \mathbf{x}_0)$, the function value in the new direction $2\mathbf{x}_n - \mathbf{x}_0$

if $f_3 \geq f_0$ **or** $(f_0 - 2f_n + f_3) \cdot (f_0 - f_n - \Delta f)^2 \geq \frac{1}{2} \Delta f (f_0 - f_n)^2$ **then**

 Use the old directions $\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_n$ for the next iteration and back to first step

else

 Go to the next step

end if

 Set $\mathbf{s}_r = \mathbf{x}_n - \mathbf{x}_0$ with the sub index r obtained previously

 Define $\mathbf{s}_r = \mathbf{x}_n - \mathbf{x}_0$, calculate $\lambda_{\min}$ minimizing $f(\mathbf{x}_n + \lambda_{\min} \cdot \mathbf{s}_r)$, set $\mathbf{x}_i = \mathbf{x}_0 + \lambda \cdot \mathbf{s}_r$, as the starting point of the new iteration

until $|f(\mathbf{x}_{i+1})| < 10^{-10}$ **and** $\frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{\mathbf{x}_i} < \varepsilon$

The tuning of the other parameters of the PSO algorithm for all the problems solved was a cognition learning rate (C_1) of 2.0 and a social learning rate (C_2) of 2.0. The number of generations (GEN) was set at

¹ SciPy is a Python-based ecosystem of open-source software for mathematics, science, and engineering.

100, and the population size (N) was set at 20, as suggested previously by Hu and Eberhart [19]. The reason for a smaller population size and fewer number of generations was that this significantly decreased the computing time. Moreover, the objective of the PSO method was to find a feasible region that minimized the violation constraints before starting the FTMS, which would lead to the complete solution of the problem. The pseudocode indicated in Algorithm 4.

Algorithm 4. Pseudocode of the Particle Swarm Optimization (PSO) method.

Initialization: Generate random population (N) of particles (i) of positions and velocities in the search space D

repeat

for Each particle i do

Update the particle's best position ($f(pb_i)$)

if $f(x_i) < f(pb_i)$ then

$pb_i = x_i$

end if

Update the global best position ($f(gb)$)

if $f(pb_i) < f(gb_i)$ then

$gb = pb_i$

end if

end for

Update particle's velocity and position

for Each particle i do

for Each dimension d do

$v_i = \omega_i \cdot v_i + C_1 \cdot \text{Rnd}() \cdot (pb_{i,d} - x_{i,d}) + C_2 \cdot \text{Rnd}() \cdot (gb_d - x_{i,d})$

$x_{i,d} = x_i + v_i$

end for

end for

$k = k + 1$

until $k < \text{Number of generations (GEN)}$

For each particle generated, the penalty function $P(\mathbf{x})$, Eq. (13), is evaluated, where T is the penalty parameter.

$$P(\mathbf{x}) = f(\mathbf{x}) + T[T(\mathbf{x})] \quad (13)$$

The initialization with the PSO method described previously was performed until the best particle was found, with the value of $T(\mathbf{x})$ (the constraint violation) being less than an arbitrary value (set at 10^{-3}). The penalty parameter acts as indicated in Algorithm 5.

Algorithm 5. Pseudocode of FTMS-PSO.

Initialization of PSO

Particles generated

Evaluation of $P(\mathbf{x})$

$k = 1.0 \times 10^{-6}$

if $[T(\mathbf{x})] \leq 10^{-3}$ then

The search continue with FTMS

else

if $T < 10^{-3}$ then

$T = T \times 10.0$

else

$T = T \times 2.0$

end if

end if

When the best point satisfied this criterion, the FTMS was started as indicated previously without any modification. This code was performed for 20 runs for each problem. The objective of using PSO hybridized with FTMS was to allow the optimization to explore different routes through the solution space and might help identify alternative solutions.

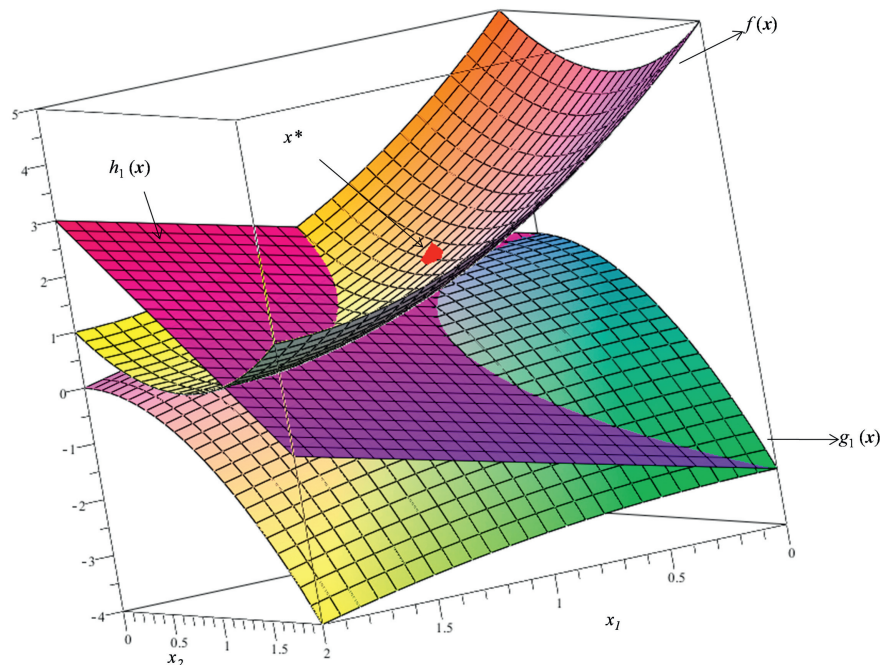


Fig. 2. Objective function ($f(\mathbf{x})$), equality constraint ($h_1(\mathbf{x})$), inequality constraint ($g_1(\mathbf{x})$) and optimum point (x^*) for the test problem of Eq. (14).

3. Results

A comparison of the performance of the optimization methods proposed in this work was conducted using a simple problem (a convex programming problem) with two variables, one equality constraint and one inequality constraint, as described in Eq. (14), [12], with the purpose of allowing the visualization of the search direction followed by the methods proposed until the optimum is found. Fig. 2 shows the objective function, the equality and inequality constraints and the optimum point.

$$\begin{aligned} \text{Minimize } & f(\mathbf{x}) = (x_1 - 2)^2 + (x_2 - 1)^2 \\ \text{Subject to: } & h_1(\mathbf{x}) = x_1 - 2x_2 + 1 = 0 \\ & g_1(\mathbf{x}) = -\frac{x_1^2}{4} - x_2^2 + 1 \geq 0 \end{aligned} \quad (14)$$

Table 1 shows the number of objective function evaluations, the number of iterations and the processing time (using the machine described previously) for the five optimization codes implemented in this work. Fig. 3 shows the trajectory of each method from the initial (near-feasible) point until the convergence to the optimum point. The initial nonfeasible point was $\mathbf{x}^{(0)} = [2 \ 2]^T$ with objective function value of $f(\mathbf{x})^{(0)} = 1.0$ for the FTM, FTMS, FTMS-BFGS and FTMS-Powell methods. The FTMS-PSO method generated the initial point randomly, and the FTMS started in the best particle founded after 100 generations, that is $\mathbf{x}^{(0)} = [0.841 \ 0.903]^T$, with the objective function value of $f(\mathbf{x})^{(0)} = 1.352$. The optimum value for this problem was $\mathbf{x}^* = [0.823 \ 0.911]^T$ when the objective function assumed the value of $f(\mathbf{x})^* = 1.393$.

Table 1
Results summary for the test problem described in Eq. (14)

	FTM	FTMS	FTMS-BFGS	FTMS-Powell	FTMS-PSO [ⓐ]
Number of objective function evaluation	50	28	45	29	22
Number of iterations	18	12	24	15	9
Processing time (seconds)	0.4913	0.0410	0.2565	0.3798	3.4341

[ⓐ] Best performance in 20 runs.

As can be observed in Table 1, the algorithms tested were able to find the optimum point for this simple problem. Comparing the number of objective function evaluations and iterations, the original FTM and the FTMS-BFGS methods presented the worst performance, with the largest values. The best performance was presented by the FTMS and FTMS-PSO methods. The FTMS showed the shortest processing time and the low numbers of iterations and objective function evaluations. For the FTMS-PSO method, the processing time was longer, due to the time consumed by the initialization step with the PSO method, however this method presented lower number of iterations and evaluations of objective function. The trajectories followed by each method (Fig. 3) indicated that FTMS and FTMS-PSO had a more direct path than the other methods. FTMS-BFGS showed search path in which the objective function reached high values (distant from the optimum point, $f(\mathbf{x}) \approx 7.165$), hence requiring a greater number of iterations and objective function evaluations in order to reach the optimal point.

3.1. Benchmark problems

Tables 2 and 3 show the number of variables, the type of objective function, the number of equality and inequality constraints, the upper and lower boundary constraints, and brief descriptions for the benchmark problems used in this work. The results were divided into two categories: (i) problems with small dimensions ($2 \leq n \leq 6$), and (ii) problems with larger dimensions ($n > 6$). As can be observed in problem descriptions, these benchmark problems cover a wide range of application including heat exchanger design (g10), chemical equilibrium (g14), maximization of the profit of hypothetical wood-pulp plant (g16), and minimization of the costs of blending multicomponent mixtures (g20), among other engineering problems. There are some generated problems important to represent the mathematical formulation and numeric issues commonly found in real constrained optimization problems. Two problems of the benchmark, g10 and g14, were selected to detailed analysis including mathematical formulation, and are described in the next section.

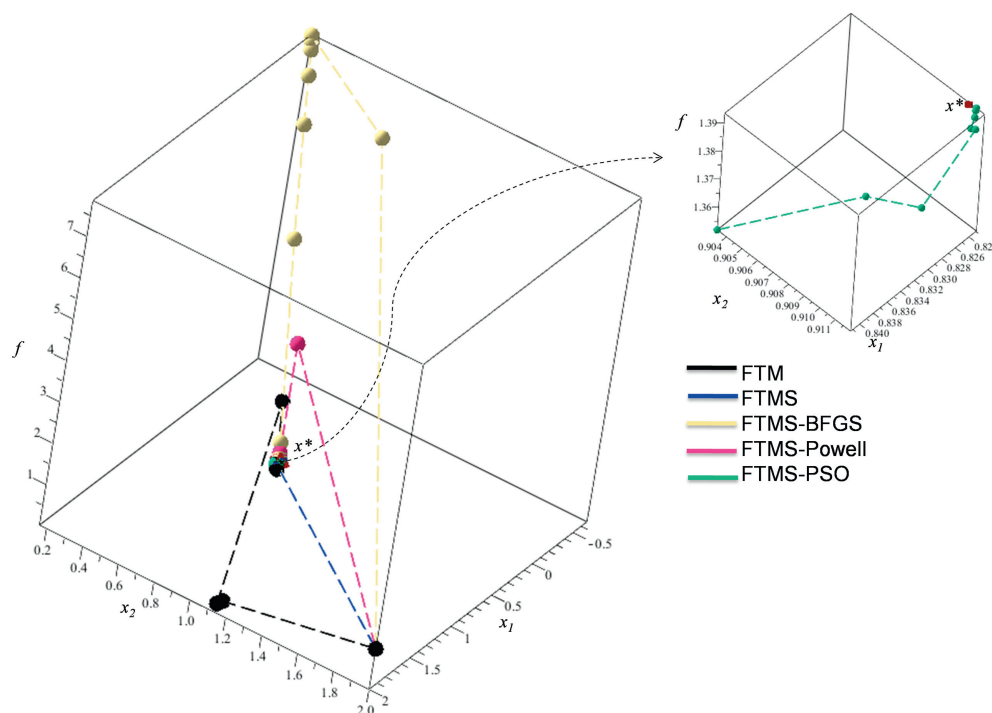


Fig. 3. Trajectories from the nonfeasible initial vector obtained using the different optimization methods for test problem (Eq. (14)). The path followed by FTMS-PSO is shown separately due to the proximity of starting point from the optimal point.

Table 2

Description of benchmark problems. Where: EC: equality constraints; IC: inequality constraints; ULBC: upper and lower bound constraints

Function	<i>n</i>	Type	EC	IC	ULBC	Description
<i>Small problems</i>						
g06	2	Cubic	0	2	4	Problem with cubic objective function and quadratic constraints.
g08	2	Nonlinear	0	2	4	Nonlinear problem with many local optima, the highest peaks are located along <i>x</i> axis. In the feasible region, the problem presents two maxima of almost equal fitness of value of 0.1.
g11	2	Quadratic	1	0	4	Problem with quadratic objective function and quadratic constraint.
g12	3	Quadratic	0	1	4	Problem with disjointed components, the feasible region of the search space consists of 9 ³ disjointed spheres.
g15	3	Quadratic	2	0	6	Nonlinear problem with nonlinear equality constraints.
g05	4	Cubic	3	2	8	Problem with cubic objective function and nonlinear constraints.
g04	5	Quadratic	0	6	10	Standard randomly generated test problem of nonconvex quadratic objective function and constraints
g13	5	Nonlinear	3	0	10	Problem with nonlinear objective function and constraints.
g16	5	Nonlinear	0	38	10	The objective function is the net profit of a hypothetical wood-pulp plant. The constraints include the usual material and energy balances as well as several empirical equations.
g17	6	Nonlinear	4	0	12	Optimization of an electrical network.

Table 3

Description of benchmark problems. Where: EC: equality constraints; IC: inequality constraints; ULBC: upper and lower bound constraints

Function	<i>n</i>	Type	EC	IC	ULBC	Description
<i>Larger problems</i>						
g09	7	Polynomial	0	4	14	Nonlinear problem with nonlinear constraints
g10 ^①	8	Linear	0	6	16	Heat exchanger design.
g18	9	Quadratic	0	13	18	Maximization the area of a hexagon in which the maximum diameter was unity.
g07	10	Quadratic	0	7	20	Problem with quadratic objective function and linear and nonlinear constraints.
g14 ^①	10	Nonlinear	3	0	20	Problem of chemical equilibrium at constant temperature and pressure.
g01	13	Quadratic	0	9	26	Standard randomly generated test problem of nonconvex quadratic programming.
g19	15	Nonlinear	0	5	30	Problem formulated by the Shell Development Company for the original Colville study.
g02	20	Nonlinear	0	2	40	Nonlinear problem with global maximum unknown.
g03	20	Polynomial	1	0	20	Problem with polynomial objective function and quadratic equality constraint.
g20	24	Linear	14	6	48	Minimization of the cost of blending multicomponent mixtures.

^① Detailed mathematical modeling and optimization formulation are showed in Section 3.2.

The results obtained after applying the optimization methods described previously (see Section 2) to the set of benchmark problems (from [13]) are described in Tables 4, 5 and 6.

The success rates² of the methods were 80% for FTM, 100% for FTMS, 75% for FTMS-BFGS, 95% for FTMS-Powell and 85% for FTMS-PSO.

Tables 4 and 5 show the optimum values found by the different optimization methods. For the first category of problems ($2 \leq n \leq 6$), it can be seen that FTMS-PSO and FTM presented the worst performance among the methods, with some unsolved problems. The best method was FTMS, which solved all the problems with good agreement. The other methods reached the convergence for almost all the problem in this category ($2 \leq n \leq 6$), and were able to reach (or arrive close to) the known optima.

The agreement with known results for problem g12 was a little poor. This problem presents multiple disjointed regions and represents a challenge for any optimization method. The optima point showed by the methods proposed here for problem g12 stays at $f(\mathbf{x})^* = -0.85$, and the optimum known is $f(\mathbf{x})^* = -1.00$. Meanwhile, even using FTMS-PSO, that was run 20 times and thus the search started from different 20 initial points, the result converged to (or near to) the same point reported here.

The problems g05 and g17 have trigonometric functions in their equality constraints. This type of functions appeared to destabilize the swarm of particles, and the PSO was not able to reach feasible start points in both problems.

For the second category of problems ($n > 6$), FTMS-BFGS showed the worst performance, with three unsolved problems, and FTMS and FTMS-PSO presented the best performance. The FTMS, FTMS-Powell and FTMS-PSO methods were able to reach convergence, with the solutions found by FTMS and FTMS-PSO providing better fits to the known optima.

Problem g19 had a nonlinear objective function with 15 independent variables and 5 cubic constraints; the 15 variables also had lower bounds.

The FTM, FTMS, FTMS-BFGS and FTMS-Powell converged very slowly to a local optimal solution. FTMS-PSO showed a good performance and converged more rapidly to the global optimum known in fewer number of iterations and objective function evaluations than the other methods. This problem clearly presents a large number of local optima, due to the various local optimal points founded by the codes tested. This problem is a challenge for any direct method of optimization (since they are more prone to be stuck in a local optimum), and the hybridization of FTMS with PSO proved to be efficient to overcome this awkwardness.

Problem g16 presents a complex objective function with a large number of nonlinear constraints. For this problem, only the FTMS was able to find the solution. The other methods do not converged to a feasible solution. This same problem is reported in literature, [12], and even indirect method (GRG) failed to solve it, probably due to difficulty in handling the first and second derivatives that are liable to human error, despite the large time required to problem preparation. It is important to highlight that FTMS was able to solve this problem in few evaluations of objective function and iterations, moreover the preparation time was quite small.

The number of objective function evaluations and the number of iterations for each method are indicated in Table 6.

Analysis of the first group of problems showed that for problems with the same number of variables, the number of evaluations could vary widely, probably due to the topology of the objective and constraint functions. Furthermore, even for a small number of variables, the number of objective function evaluations could reach high values, as in the case of problem g17 solved by FTMS. This problem (an electrical network optimization), which has a nonlinear objective function defined by parts and nonlinear equality constraints, was solved by FTMS after 10295 objective function evaluations and 71 iterations. For this same problem, the FTMS-Powell method reached the optimum after fewer objective function evaluations (431) but with a larger number of iterations (133).

The FTMS, FTMS-Powell and FTMS-PSO methods presented the best performance for the second group of problems, which were all solved, with FTMS and FTMS-PSO providing better agreement with the known optima. The other methods (FTM and FTMS-BFGS) presented

² In this work, the success rate is defined as the number of problems solved divided by the number of problems proposed for resolution (g01 to g20).

Table 4

Optimum known and found by the algorithms for the benchmark problems

Function	n	Optimum known	Optimum calculated					
			FTM	FTMS	FTMS-BFGS	FTMS-Powell	FTMS-PSO	
							Best	Worst
Small problems								
g06	2	−6961.814	−6962.275	−6961.681	−6959.518	−6976.961	−6961.814	−6961.032
g08	2	0.100	0.096	0.096	0.096	0.096	0.096	0.094
g11	2	0.750	0.753	0.750	0.749	0.749	0.750	0.749
g12	3	−1.000	−0.859	−0.859	−0.840	−0.843	−0.859	−0.849
g15	3	961.715	961.716	961.715	961.715	961.715	961.716	961.715
g05	4	5126.498	5126.499	5126.653	5126.498	5126.498	−	−
g04	5	−30665.609	−	−32217.421	−32214.312	−32196.566	−32217.431	−32217.431
g13	5	0.054	0.0546	0.054	0.438	0.054	0.054	0.439
g16	5	−1.905	−	−1.904	−	−	−	−
g17	6	8853.540	−	8853.649	−	8853.540	−	−

Table 5

Optimum known and found by the algorithms for the benchmark problems

Function	n	Optimum known	Optimum calculated					
			FTM	FTMS	FTMS-BFGS	FTMS-Powell	FTMS-PSO	
							Best	Worst
Larger problems								
g09	7	680.630	681.873	680.630	680.687	687.225	680.630	687.645
g10	8	7049.331	–	7050.925	–	9623.854	7049.308	7095.352
g18	9	–0.866	–0.859	–0.866	–	–0.847	–0.866	–0.499
g07	10	24.306	24.083	24.120	78.069	14865.964	24.331	21.900
g14	10	–47.765	–47.645	–47.443	–47.761	–47.761	–47.759	–47.549
g01	13	–15.000	–15.000	–15.000	–11.700	–11.700	–15.000	–11.074
g19	15	32.656 ^① 32.386 ^②	21.115	44.321	148.694	350.348	32.343	–0.477
g02	20	–0.804	–0.787	–0.793	–	–0.781	–0.379	–0.232
g03	20	1.000	0.986	0.997	1.000	0.881	–0.999	–0.776
g20	24	0.205 ^① 0.057 ^②	0.0586	0.0577	0.0567	0.0576	0.0567	0.0583

^① Optimum value reported by [13].^② Optimum value reported by [12].**Table 6**

Number of objective function evaluations and iterations (in [])

Function	n	FTM	FTMS	FTMS-BFGS	FTMS-Powell	FTMS-PSO	
						Best	Worst
Small problems							
g06	2	45 [40]	22 [37]	49 [25]	3554 [4]	21 [11]	22 [12]
g08	2	77 [40]	66 [37]	69 [36]	106 [58]	55 [30]	72 [37]
g11	2	75 [19]	185 [77]	244 [126]	266 [133]	76 [19]	271 [133]
g12	3	287 [137]	104 [31]	87 [48]	129 [53]	100 [31]	232 [107]
g15	3	154 [62]	176 [80]	195 [81]	268 [123]	107 [45]	202 [96]
g05	4	2931 [739]	36019 [1531]	736 [373]	111 [53]	–	–
g04	5	–	573 [329]	663 [352]	727 [396]	261 [112]	964 [518]
g13	5	130 [52]	242 [120]	315 [158]	279 [140]	208 [98]	243 [115]
g16	5	–	369 [146]	–	–	–	–
g17	5	–	10,295 [71]	–	431 [133]	–	–
Larger problems							
g09	7	2401 [1614]	862 [558]	748 [418]	1270 [729]	580 [364]	572 [359]
g10	8	–	734 [358]	–	1861 [964]	2035 [1203]	763 [356]
g18	9	1704 [1025]	784 [445]	–	823 [428]	655 [388]	737 [433]
g07	10	2504 [1765]	2345 [1636]	1209 [723]	972 [569]	2440 [1722]	2423 [1722]
g14	10	1492 [860]	1613 [924]	1515 [1001]	1643 [1023]	1328 [786]	850 [460]
g01	13	592 [331]	476 [290]	1752 [895]	3294 [1639]	339 [194]	2769 [1597]
g19	15	3109 [2081]	2558 [1892]	2289 [1437]	2669 [1703]	2231 [1577]	1965 [1289]
g02	20	2342 [1413]	2320 [1513]	–	2533 [1911]	2623 [1994]	2625 [1994]
g03	20	2578 [1640]	1962 [1239]	2663 [1640]	886 [422]	2427 [1562]	2665 [1718]
g20	24	1899 [675]	2200 [951]	3648 [1808]	2746 [1378]	2503 [1075]	2478 [1110]

similar performance in terms of the numbers of iterations and objective function evaluations.

In the case of problem g20, which is complex and represents a challenge for any nonlinear programming method, the results found with the codes tested in this work were in agreement with those ones reported elsewhere [12].

FTMS-PSO showed a good performance compared with the other methods. This code can be useful when the optimization problem has an unknown starting point, or when the known starting point leads to an unfeasible solution. Numerical experiments also indicated improvements in the capacity of a direct method (as FTMS) escape from local optima when hybridized with PSO, that generate different starting points for the deterministic method of search.

3.2. Detailed study cases

3.2.1. g10 – heat exchanger design

This problem was first presented by Avriel and Williams [20], and have been employed by many authors since then during optimization tests, as Dembo [21], Teles *et al.* [22], and Andrei [23], among others. This study case deals with the minimization of the total area of a net of heat exchangers with non-convexities as bilinear terms from energy balances, which brings great difficult to optimization methods. As indicated in Fig. 4, the fluid with flow rate W and specific heat C_p ($WC_p = 10^5 \text{ Btu} \cdot \text{h}^{-1} \cdot ^\circ\text{F}^{-1}$, $1 \text{ Btu} = 1.055 \text{ kJ}$, $t/^\circ\text{C} = 5(t/^\circ\text{F} - 32)/9$) need to be heated from temperature T_0 (100°F) to T_3 (500°F) using three hot streams with different inlet temperatures ($T_{H1,\text{in}} = 300^\circ\text{F}$, $T_{H2,\text{in}} = 400^\circ\text{F}$ and $T_{H3,\text{in}} = 600^\circ\text{F}$) and the overall transfer coefficients of the exchangers $U_1 = 120$, $U_2 = 80$, $U_3 = 40 \text{ Btu} \cdot \text{h}^{-1} \cdot \text{ft}^{-2} \cdot ^\circ\text{F}^{-1}$ ($1 \text{ Btu} = 1.055 \text{ kJ}$, $1 \text{ ft} = 0.3048 \text{ m}$, $t/^\circ\text{C} = 5(t/^\circ\text{F} - 32)/9$) are constants.

The mathematical formulation can be written as (Avriel and Williams [20], Liang *et al.* [13] and Andrei [23]):

$$\text{Minimize : } A_1 + A_2 + A_3 \quad (15)$$

Subject to:

$$\begin{aligned} T_1 + T_{H1,\text{out}} - T_0 - T_{H1,\text{in}} &\leq 0 \\ -T_1 + T_2 + T_{H2,\text{out}} - T_{H1,\text{in}} &\leq 0 \\ T_{H3,\text{out}} - T_2 - T_{H3,\text{in}} + T_3 &\leq 0 \\ A_1 - A_1 T_{H1,\text{out}} + \frac{WC_p}{U_1} T_1 - \frac{WC_p}{U_1} T_0 &\leq 0 \\ A_2 T_1 - A_2 T_{H2,\text{out}} - \frac{WC_p}{U_2} T_1 + \frac{WC_p}{U_2} T_2 &\leq 0 \\ A_3 T_2 - A_3 T_{H3,\text{out}} - \frac{WC_p}{U_3} T_2 + \frac{WC_p}{U_3} T_3 &\leq 0 \\ 100 \leq A_1 \leq 10000 \\ 1000 \leq A_2, A_3 \leq 10000 \\ 10 \leq T_1, T_2, T_{H1,\text{out}}, T_{H2,\text{out}}, T_{H3,\text{out}} &\leq 1000 \end{aligned}$$

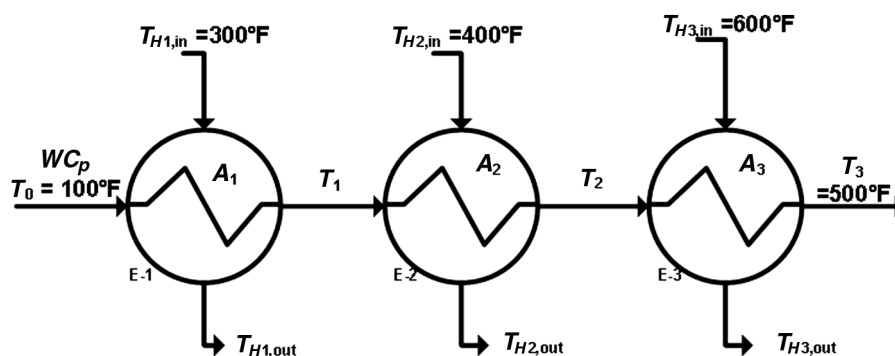


Fig. 4. Three stage heat exchanger system ($t/^\circ\text{C} = 5(t/^\circ\text{F} - 32)/9$). Adapted from Avriel and Williams [20].

The optimization methods, FTM and the other methods proposed in this work (FTMS, FTMS-BFGS, FTMS-Powell and FTMS-PSO) were applied to obtain the solution of the problem. The initial (not feasible) point is $x_0 = [A_1, A_2, A_3, T_1, T_2, T_{H1,\text{out}}, T_{H2,\text{out}}, T_{H3,\text{out}}]$ was employed by FTM, FTMS, FTMS-BFGS and FTMS-Powell, where $x_0 = [5000.0, 5000.0, 5000.0, 200.0, 350.0, 150.0, 225.0, 425.0]$. This point was not used by FTMS-PSO, since the starting point was found by stochastic method PSO, as described previously.

As indicated in Tables 5 and 6, the original method FTM was not able to solve the problem, as well as the scaling method hybridized with BFGS (FTMS-BFGS). The FTM method with scaled variables (FTMS) and its hybridization with Powell (FTMS-Powell) and PSO (FTMS-PSO) was able to find an optimum. The solution with areas and temperatures found by each method is showed in Fig. 5. The known optimum from literature is also indicated.

From Fig. 5 and Table 5 it can be observed that FTMS and FTMS-PSO reach the best values and agree with the data reported in literature. The total area for the heat exchangers reported in literature [13] was 7049.331 ft^2 , while FTMS found 7050.925 ft^2 , FTMS-Powell 9623.854 ft^2 and FTMS-PSO_{Best} 7049.308 ft^2 . The FTMS-PSO_{Best} showed slightly better optimum value than the one reported by Liang *et al.* [13]. Although the FTMS-Powell has found an optimum, the value is much larger than the optimum known and could be associated with a local optimum. The distribution of areas among the heat exchangers presented some small differences as can be noted in Fig. 5.

Analyzing the computational effort (Table 6) of the best methods, the result obtained with FTMS used 734 objective function evaluations, 358 iterations in 26.14 s, while FTMS-PSO_{Best} employed 2035 objective function evaluations, 1203 iterations in 64.15 s. This same problem was solved by Teles *et al.* [22] using GAMS 23.2 with CPLEX 12.1 as the MILP solver. The authors employed a multi-parametric disaggregation technique for polynomial programming problems, and reported the value of objective function of 7049.248022 ft^2 using 6 significant digits in parametrization, and took 13 min to solve the problem.

The results found in this work employing simple optimization methods, FTMS and FTMS-PSO, demonstrate the efficiency of the proposed methods to lead with difficult problems, since the original method FTM did not solve the problem. Besides, the methods also have a good performance with small processing time.

3.2.2. g14 – chemical equilibrium

This problem deals with the determination of the equilibrium composition resulting from combustion of a stoichiometric mixture of hydrazine and oxygen ($\frac{1}{2} \text{N}_2\text{H}_4 + \frac{1}{2} \text{O}_2$) at a temperature of 3500 K and pressure of 750 psi (1 psi = 6894.76 Pa). This problem was proposed by White *et al.* [24], and was used since then to study methods of chemical equilibrium calculations using different optimization techniques, as reported by

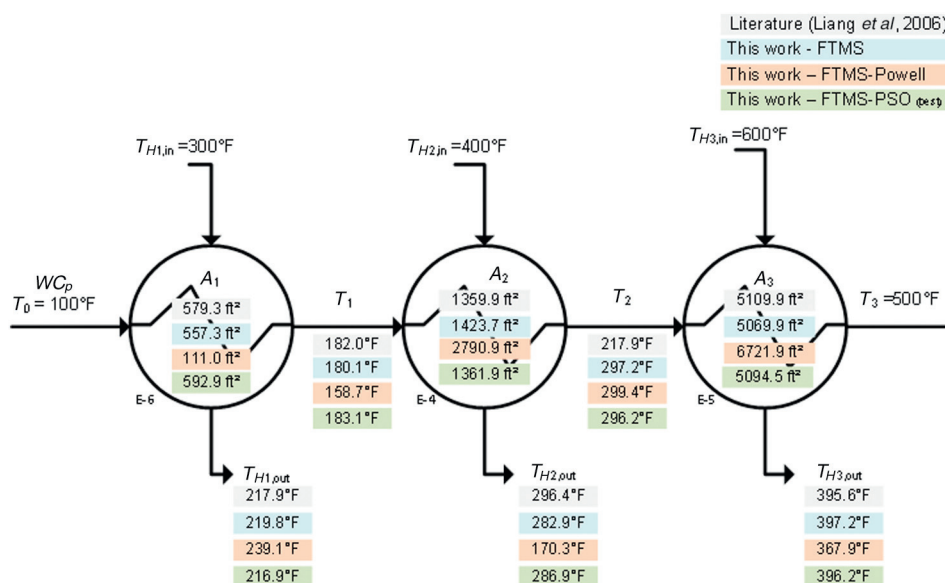


Fig. 5. Optimization solution of the three stage heat exchanger system with different methods (1ft = 0.3048m, $t/^{\circ}\text{C} = 5(t/^{\circ}\text{F} - 32)/9$).

Bracken and McCormick [25], Blečić *et al.* [26], and Rangaiah [27], among others.

Hydrazine and its derivatives have been used as rocket propellants ([28,29]), and the knowledge of thermodynamic equilibrium of combustion products is an important issue, since a well designed combustion chamber is very nearly of thermodynamic equilibrium [30]. As reported by Lima *et al.* [31], chemical equilibrium in reactions is fundamental in reactor design, and also shows the most favorable conditions for the occurrence of a certain reaction and can predict the composition in equilibrium state.

The problem formulation is the minimization of the total free energy of the mixture with 10 components (indicated in Table 7) subject to mass balance relationships for the elements. The mathematical formulation can be written as (White *et al.* [24], Bracken and McCormick [25], Liang *et al.* [13]):

$$\text{Minimize: } f(\mathbf{x}) = \sum_{i=1}^{10} x_i \left(c_i + \ln \frac{x_i}{\sum_{j=1}^{10} x_j} \right) \quad (16)$$

Subject to:

$$h_1(\mathbf{x}) = x_1 + 2x_2 + 2x_3 + x_6 + x_{10} - 2 = 0$$

$$h_2(\mathbf{x}) = x_4 + 2x_5 + x_6 + x_7 - 1 = 0$$

$$h_3(\mathbf{x}) = x_3 + x_7 + x_8 + 2x_9 + x_{10} - 1 = 0$$

$$0 \leq x_i \leq 10$$

Table 7

Data on $1/2\text{N}_2\text{O}_4 + 1/2\text{O}_2$ at 3500 K and 750 psi (1psi = 6894.76Pa)
Source: Bracken and McCormick [25].

i	Compound	c_i
1	H	-6.089
2	H ₂	-17.164
3	H ₂ O	-34.054
4	N	-5.914
5	N ₂	-24.721
6	NH	-14.986
7	NO	-24.100
8	O	-10.708
9	O ₂	-26.662
10	OH	-22.179

where x_i is the number of moles of the 10 compounds present in the equilibrium mixture, c_i is given by Eq. (17) [25] and the values are indicated in Table 7, where (F^0/RT) is the modal standard (Gibbs) free energy function for the i th compound and P is the total pressure in atmospheres.

$$c_i = \left(\frac{F^0}{RT} \right)_i + \ln P \quad (17)$$

This problem has a nonlinear objective function with a logarithmic function of ten independent variables subject to three linear equality constraints, and the independent variables had zero lower bounds. The optimization results for mixture composition in equilibrium are indicated in Table 8 for all methods, the original FTM and the proposed ones FTMS, FTMS-BFGS, FTMS-Powell and FTMS-PSO.

The solution of this problem was achieved by all methods as indicated in Table 8, and the value of objective function found by all methods is very close to the known optimum. The method that presented the best performance was the FTMS-PSO, comparing the objective function value and the equilibrium composition with known values reported in literature. This method also presented a superior performance regarding the number of objective function evaluations (1328) and iterations (786), when compared with the other methods. Except the composition of the mixture found by FTM and FTMS, the other methods found a similar equilibrium composition, which corroborates to the global optimum known.

4. Conclusions

This paper proposes the use of the flexible tolerance method (FTM) with scaling of variables and hybridization with different unconstrained methods (BFGS and modified Powell) to perform the inner search and with a stochastic method (PSO) to perform the initialization. The FTMS and FTMS-PSO method proved to be the most effective when applied to the benchmark problems employed. Hybridization with other unconstrained optimization methods (BFGS and Powell) did not result in any further enhancement of performance.

The flexible tolerance method with scaling (FTMS) was more efficient and computationally more economical than the original flexible tolerance method (FTM). Furthermore, the method required little tuning of parameters, and the same parameter configuration was able

Table 8

Optimization results for problem g14

Variable	Optimum known [13]	FTM	FTMS	FTMS-BFGS	FTMS-Powell	FTMS-PSO _{Best}
$f(\mathbf{x})$	−47.764	−47.645	−47.443	−47.443	−47.761	−47.759
<i>Mixture composition at equilibrium</i>						
H (x_1)	0.0407	0.0837	0.1189	0.0395	0.0407	0.0404
H ₂ (x_2)	0.1477	0.2479	0.1517	0.1489	0.1478	0.1506
H ₂ O (x_3)	0.7832	0.6851	0.7602	0.7818	0.7831	0.7783
N (x_4)	0.0014	0.0029	0.0093	0.0016	0.0014	0.0034
N ₂ (x_5)	0.4853	0.4835	0.4816	0.4847	0.4852	0.4814
NH (x_6)	0.0007	0.0002	0.0201	0.0007	0.0007	0.0015
NO (x_7)	0.0270	0.0299	0.0075	0.0283	0.0274	0.0322
O (x_8)	0.0180	0.0110	0.1459	0.0181	0.01794	0.0180
O ₂ (x_9)	0.0373	0.1120	0.0246	0.0367	0.03733	0.0356
OH (x_{10})	0.0969	0.0501	0.0372	0.0984	0.09688	0.1001

to solve all the problems in good agreement with the known solutions, with a few exceptions (such as the problem with disjointed regions, g12). The FTMS employed the Nelder–Mead method to perform the inner search, as originally proposed by Paviani and Himmelblau [2], and used scaled variables as proposed in this work.

The flexible tolerance method with scaling (FTMS) and hybridized with the stochastic method of particle swarm (PSO), FTMS-PSO, presented some advantages applied to the solution of the nonlinear optimization problems. The relative advantages of the methods (FTMS and PSO) could be employed, and the stochastic method PSO provided good initial points for the deterministic method FTMS and helped escape from local optima. This strategy was useful to corroborate the global optimum in nonlinear optimization problems, which the global optimum cannot be guaranteed.

Typical chemical engineering problems were detailed, heat exchanger design (g10) and chemical equilibrium (g14), and the optimization methods proposed presented a superior performance when compared with the original method (FTM) and other methods presented in literature. For these problems (g10 and g14), FTMS and FTMS-PSO also showed better results than the other methods proposed, in accordance with the results found for the other problems of the benchmark analyzed.

Acknowledgments

The authors thank CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) and CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico, grant number 161464/2013-0) for the financial support.

References

- [1] R. Smith, Chemical Process: Design and Integration, Wiley, 2005.
- [2] D. Paviani, D. Himmelblau, Constrained nonlinear optimization by heuristic programming, *Oper. Res.* 17 (5) (1969) 872–882.
- [3] J.M. Martínez, F.N.C. Sobral, Constrained derivative-free optimization on thin domains, *J. Glob. Optim.* 56 (3) (2013) 1217–1232.
- [4] R. Fenton, W. Cleghorn, J. Fu, A modified flexible tolerance method for nonlinear programming, *Eng. Optim.* 15 (2) (1989) 141–152.
- [5] P. Zhang, Y. Ren, Flexible tolerance simplex method and its application to multicomponent spectrophotometric determinations, *Anal. Chim. Acta* 222 (1) (1989) 323–333.
- [6] D. Constantinescu, Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, *J. Robot. Syst.* 17 (2000) 223–249.
- [7] X. Chen, Y. Yin, The flexible tolerance method for estimating hydrologic parameters in the root zone, *J. Am. Water Resour. Assoc.* 42 (2) (2006) 495–512.
- [8] W. Shang, S. Zhao, Y. Shen, A flexible tolerance genetic algorithm for optimal problems with nonlinear equality constraints, *Adv. Eng. Inform.* 23 (3) (2009) 253–264.
- [9] S. Omowunmi, A. Susu, Application of tikhonov regularization technique to the kinetic data of an autocatalytic reaction: Pyrolysis of n-eicosane, *Engineering* 3 (2011) 1161–1170.
- [10] A. Lima, W. Kwong, A. Cruz, Comparative study and application of the flexible tolerance method in synthesis and analysis of processes with mass integration, *J. Chem. Chem. Eng.* 7 (3) (2013) 228–238.
- [11] M. Naish, Sensing-system planning for the surveillance of moving objects Ph.D. thesis, Department of Mechanical and Industrial Engineering, University of Toronto, Canada, 2004.
- [12] D. Himmelblau, Applied Nonlinear Programming, McGraw-Hill, USA, 1972.
- [13] J. Liang, T. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. Coello, K. Deb, Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization, Tech. rep, Nanyang Technological University, Singapore, 2006.
- [14] P. Gill, W. Murray, M. Wright, Practical Optimization, Academic Press, USA, 1981.
- [15] J. Nocedal, S. Wright, Numerical Optimization, Springer series in operations research and financial engineering, Springer, Netherland, 1999.
- [16] E. Jones, T. Oliphant, P. Peterson, et al., SciPy: Open source scientific tools for Python, 2001 URL.
- [17] M.J.D. Powell, An efficient method for finding the minimum of a function of several variables without calculating derivatives, *Comput. J.* 7 (2) (1964) 155–162.
- [18] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, C. Gagné, DEAP: Evolutionary algorithms made easy, *J. Mach. Learn. Res.* 13 (2012) 2171–2175.
- [19] X. Hu, R. Eberhart, Solving constrained nonlinear optimization problems with particle swarm optimization, 6th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2002) UK 2002, pp. 203–206.
- [20] M. Avriel, A.C. Williams, An extension of geometric programming with applications in engineering optimization, *J. Eng. Math.* 5 (2) (1971) 187–194.
- [21] R.S. Dembo, A set of geometric programming test problems and their solutions, *Math. Program.* 10 (1) (1976) 192–213.
- [22] J.P. Teles, P.M. Castro, H.A. Matos, Multi-parametric disaggregation technique for global optimization of polynomial programming problems, *J. Glob. Optim.* 55 (2) (2013) 227–251.
- [23] N. Andrei, Applications in Chemical Engineering, Springer, US, Boston, MA, 2013 161–222.
- [24] W.B. White, S.M. Johnson, G.B. Dantzig, Chemical equilibrium in complex mixtures, *J. Chem. Phys.* 28 (5) (1958) 751–755.
- [25] J. Bracken, G.P. McCormick, Selected Applications of Nonlinear Programming, John Wiley & Sons, Inc., New York, 1968.
- [26] J. Blečić, J. Harrington, M.O. Bowman, Tea: A code calculating thermochemical equilibrium abundances, *Astrophys. J. Suppl. Ser.* 225 (1) (2016) 4.
- [27] G.P. Rangaiah, Studies in constrained optimization of chemical process problems, *Comput. Chem. Eng.* 9 (4) (1985) 395–404.
- [28] H. Helvajian, Microengineering Aerospace Systems, Aerospace Press, USA, 1999.
- [29] S.M. Davis, N. Yilmaz, Advances in hypergolic propellants: Ignition, hydrazine, and hydrogen peroxide research, *Adv. Aerosp. Eng.* 2014 (1) (2014) 9.
- [30] P. Hill, C. Peterson, Mechanics and Thermodynamics of Propulsion, Addison-Wesley, USA, 1992.
- [31] A.M. Lima, A.J. Assis, C.E. Hori, M.H.E. Reis, A.E.H. Machado, Thermodynamic analysis of ethanol dehydration to ethylene through equilibrium constant method using classic thermodynamics and quantum chemistry, *I.R.C.H.E.* 4 (5) (2012) 466–473.