BOHR International Journal of Smart Computing and Information Technology
2025, Vol. 6, No. 1, pp. 12–17
DOI: 10.54646/bijscit.2025.47
www.bohrpub.com

ORIGINAL RESEARCH

A Comprehensive Study of MATLAB Optimization Toolbox Solvers for Nonlinear Constraints and Objective Functions

Anup Kumar Thander[1*†] and Dwaipayan Bhowmik[2]

[1]Department of Applied Science and Humanities, Guru Nanak Institute of Technology, Kolkata, India
[2]Department of Computer Science and Engineering, Dr. Sudhir Chandra Sur Institute of Technology and Sports Complex, Kolkata, India

***Correspondence:**
Anup Kumar Thander,
anup.thander@gnit.ac.in

†ORCID:
Anup Kumar Thander,
0000-0002-6282-7076

Received: 15 January 2025; **Accepted:** 24 January 2025; **Published:** 12 February 2025

In this paper, four different solvers available in the optimization toolbox of MATLAB for nonlinear constraints and objective functions have been discussed. Among these solvers, numerical comparisons have been made using CPUTIME as the parameter. The MATLAB solvers described in this study have been applied to obtain the global optimum values of Rosenbrock's banana functions in multi-dimensions. Additionally, graphical analysis is provided for a visual illustration of the convergence of the optimal solution of Rosenbrock's function.

Keywords: MATLAB solvers, nonlinear constraints, CPUTIME, objective functions, optimal solutions

Introduction

Optimization (1–10) is a systematic technique which aims at locating better solutions, answers, or designs not easily perceptible by human intuition or trial and error. It is known as the skill of making things better. Optimization has two branches: theoretical and computational. Theoretical optimization helps in creating new methods for optimization and assessing their performance. However, computational implementation is what matters most to apply optimization successfully in real-world situations.

MATLAB is a favored calculative designing and scripting utility that offers a good and clear optimization berth. Its favor comes from its aptitude and wide use in engineering plus other areas. MATLAB Toolboxes (9) are groups of functions that help with studying and making new things in many tech spaces. They give easy but strong tools that can be reached using simple commands or visual screens. Many uses are backed by these toolboxes, which can easily fit into MATLAB programs.

This research will revolve around the use of MATLAB Toolboxes based on the optimization. It will consider four different types. To offer solvers such as *fminsearch, fmincon, fminunc*, and *lsqnonlin*, these toolboxes are evaluated for nonlinear Rosenbrock's banana functions. Numerical comparisons are made taking into account the time taken by each solver in terms of computational time (CPUTIME) (11, 12) as it directly reflects the measurement of efficiency of computations: number of iterations, complexity of algorithm, and performance of hardware. Any performance aspect like number of iterations, memory consumed, etc., is not primary in this case, taking a secondary role due to wide-ranging costs per iteration and scientific resource availability. Here optimal solutions for multidimensional Rosenbrock's banana functions are obtained through these MATLAB solvers. Convergence of optimal solutions will also be graphically demonstrated. All scientific computation was done in MATLAB because of its extensive library of toolboxes, inbuilt prototyping environment, and advanced visualization tools. Community support is huge, and their integration with languages surges the scale of work that

one can perform with MATLAB. These backings make it a suitable platform for advancing and analyzing optimization problems in research.

In short, this paper deals with MATLAB Toolboxes use in optimizing problems. It compares different solvers in the Optimization Toolbox. It is also about nonlinear Rosenbrock's banana functions, and the paper provides computational time and numerical comparisons, as well as graphs of convergence for optimal solutions.

Different types of optimization solvers

To calculate the minimum value of a function in a bounded interval, MATLAB offers the solver *fminbnd*. For nonlinear constraint problems with objective functions, the solver *fgoalattain* is the right choice. On problems that have linear constraints, goals, and integer variables, we use the solver *intlinprog*. Solver *lsqnonneg* solves linear least-squares problems in a nonnegative way (13).

fminsearch is used to search for the best solution of an unsupervised problem. This method utilizes a basic simplex algorithm based on Nelder and Mead (14), which is a geometric search method that does not make use of derivatives of the main objective function. The iterative update of the simplex allows for functional optimization. It employs four geometrical transformations—reflection, expansion, contraction, and shrinkage—to converge on a minimum point. This derivative-free method adapts based on function evaluations. On other side, the *fminunc* function is considered for unconstrained problems and utilizes a derivative-based algorithm. It endeavors to approximate both the first derivative and the Hessian matrix, which represents the second derivatives. It is recommended to use *fminunc* for improved efficiency and faster convergence to the optimal solution compared to *fminsearch* (9). When a gradient is provided, *fminunc* requires a fewer number of iterations (see (15)). The "trust-region" algorithm within *fminunc* is often faster and consumes less memory compared to the "quasi-newton" algorithm (11–13, 16–19).

The preferred solver for nonlinear sum of squares problems is *lsqnonlin*. It offers higher efficiency compared to *fminunc*, particularly when a gradient is not provided. Instead of expressing the objective as a sum of squares, lsqnonlin expects the internally squared and summed vector. The optimization process and other aspects can be controlled by specifying options using optimset.

To summarize, MATLAB provides a range of solvers for different optimization scenarios. The appropriate solver depends on the specific problem characteristics and requirements. The solvers mentioned, such as *fminbnd, fgoalattain, intlinprog, lsqnonneg, fminsearch, fminunc*, and *lsqnonlin,* cater to various optimization needs, including constrained and unconstrained problems with different objectives and constraints (9, 20).

Rosenbrock's banana functions

Now, the Rosenbrock's banana function (9) is one of the most common non-convex functions in optimization. It is used as a benchmark problem for testing various optimization techniques. The function is described mathematically as follows:

$$f(x, y) = (a - x)^2 + b(y - x^2)^2 \quad (1)$$

Here the parameters a and b command the shape of the function and its optimization difficulty. Changing a shift, the valley left or right. b controls the steepness and curvature of the banana-shaped valley. Higher values for b mean that the function is even more ill-conditioned; thus, the valley narrows and elongates. Conversely, a small value for b implies it is generally easy to optimize the function. An increase in the origins the minimum to shift towards the right. With respect to function b, the ability to optimize this function decreases because the valley becomes thinner. Due to its prominence, the Rosenbrock's function serves as a benchmark problem for researchers and practitioners to compare the performance of various optimization algorithms when facing non-convex problems. The banana function of Rosenbrock's helps us compare the performance of different algorithms and analyze their convergence properties and computational efficiency (9).

The Rosenbrock's banana function has a global minimum at $(x,y) = (a,a^2)$, where $f(x,y)$ is equal to 0. In typical scenarios, parameters are often set as $a = 1$ and $b = 100$. It's worth noting that in a special case, the function becomes symmetric when $a = 0$, and the origin (0, 0) is where the optimum point is situated. Consider a problem with two variants. The first variant involves the summation of N/2 uncoupled two-dimensional Rosenbrock's problems. This particular variant [see Equation (2)] is applicable only when N is an even number.

$$f(x_1, x_2, x_3, \ldots\ldots\ldots, x_N) = \sum_{i=1}^{N/2} [k(x_{2i-1}^2 - x_{2i})^2 + (x_{2i-1} - 1)^2]. \quad (2)$$

N is the dimension which decides the number of terms related to one another. More difficult optimization problems are obtained by increasing the number N. The parameter k scales the difficulty of optimization. A higher k generated a more ill-conditioned optimization function possessing a valley surface with a quite narrow curvature. The value of k is generally set to 100.

The solutions to this version are predictably straightforward.

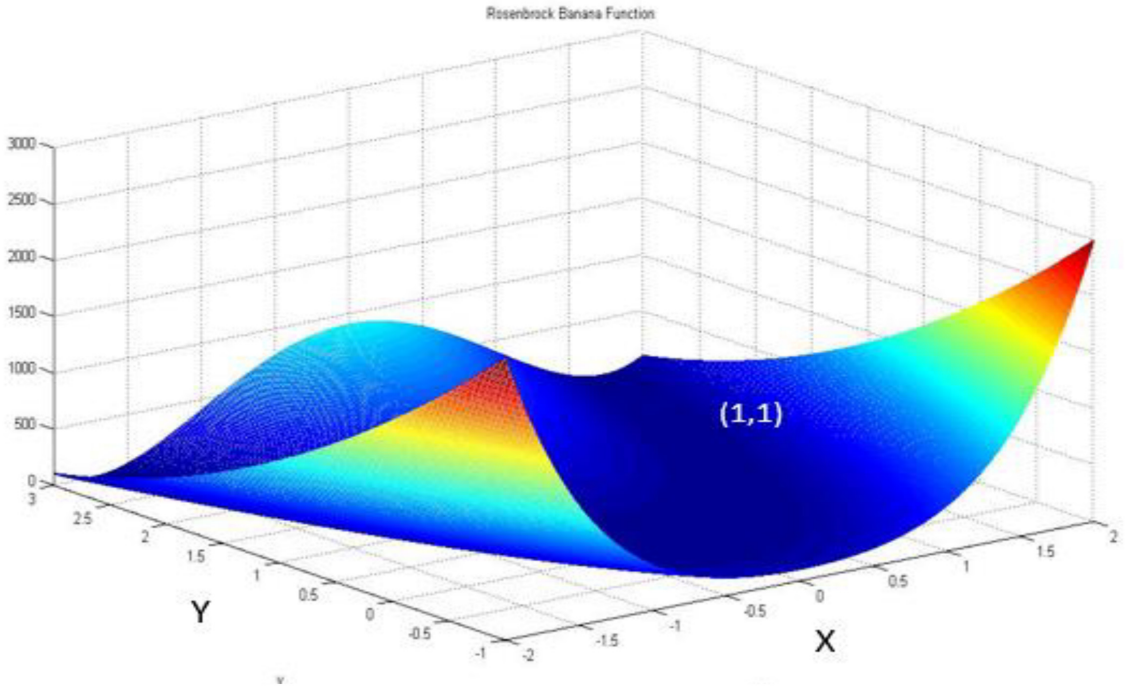

FIGURE 1 | Three-dimensional surface plot of Rosenbrock's function for $k = 100$, showing a narrow curved valley.

Another, more complex variation is

$$f(X) = \sum_{i=1}^{N-1} \left[k\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2 \right]$$
$$\text{where } X = (x_1, x_2, \ldots\ldots\ldots, x_N) \in R^N \quad (3)$$

k is constant in both cases.

The second variant exhibits specific behavior depending on the value of N. For $N = 3$, it has exactly one optimum point (minimum) located at (1, 1, 1). For N ranging from 4 to 7 (both are inclusive), there are two minima: a global minimum at (1, 1, ...,1) and a local optimum close to $\hat{x} = (-1, 1, \ldots, 1)$. These minima are determined by establishing a rational function of x by setting function's gradient to zero. To analyze the roots of the rational function, exact polynomial expressions can be derived for small values of N. Sturm's theory can then be employed to find the number of real roots within the range of $|x_i| < 2.4$. However, as N becomes larger, this approach becomes less effective due to the size of the coefficients involved. The Rosenbrock's function can be optimized efficiently by employing an adaptive coordinate system, eliminating the need for gradient information or local approximation models, unlike many derivative-free optimizers. Here all four MATLAB solvers (*fminsearch, fmincon, fminunc, and lsqnonlin)* are tested for finding the optimal solution of Rosenbrock's function in such a way that the highest number of iterative steps is $n = 10^3$ with precision $\varepsilon = 1 \times 10^{-10}$, $\delta = 1 \times 10^{-10}$. These are standard in any optimization study which ensures a balance between accuracy and efficiency. Our computer programs use the following halting conditions to ensure efficient and effective program execution (6, 7, 18).

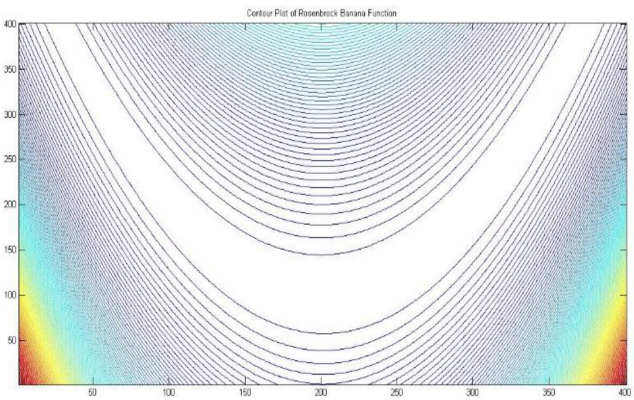

FIGURE 2 | Two-dimensional contour plot of Rosenbrock's function for $k = 100$, highlighting the steep walls and flat valley.

$$||x_{i+1} - x_i|| < \varepsilon, \quad (4)$$
$$||f(x_{i+1})|| < \delta \quad (5)$$

Also, it must be mentioned that all the solvers are used here in a system with the configuration: Intel(R) Core(TM) i5-4200U CPU @ 2.30 MHz.

Figure 1 showcases a surface plot of two-dimensional Rosenbrock's function with $k = 100$, while **Figure 2** displays a contour plot of the same function. The surface plot and contour plot of Rosenbrock's function provide insightful visualizations of its complex landscape, highlighting the challenging optimization characteristics of the function. These plots are instrumental in studying and evaluating optimization algorithms, aiding in the development of

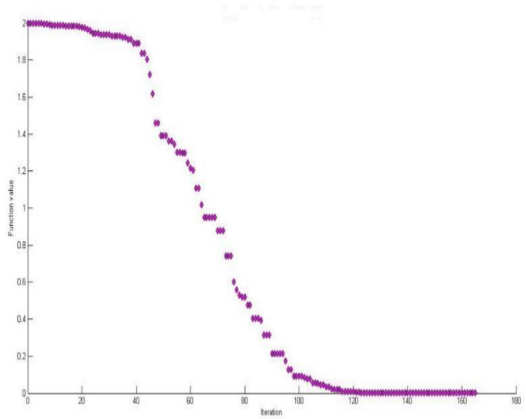

FIGURE 3 | Convergence of optimal solution of 3D Rosenbrock's banana function using MATLAB solver fminsearch, taking $x_0 = 0$, $y_0 = 0, z_0 = 0$, $k = 100$.

TABLE 1 | Numerical comparison of CPUTIME to find optimal solution by different MATLAB solvers for different values of *k*.

	fminsearch	*fminunc*	*lsqnonlin*
κ	CPUTIME	CPUTIME	CPUTIME
05	0.6864044000	0.2340015000	0.0156001000
10	0.7644049000	0.2808018000	0.0312002000
20	0.9984064000	0.2808017999	0.0468002999
30	0.9516061000	0.2496016000	0.0780004999
40	1.0140065000	0.2964019000	0.0468002999
50	0.9828063000	0.3120020000	0.0312002000
100	1.2792081999	0.3276021000	0.0780004999
200	1.3884088999	0.2964018999	0.0780005000

efficient strategies to solve optimization problems. The contour plot shows the thinner curved valley, showing difficulty in convergence. The surface plot displays the steep walls and flat valley, highlighting why gradient-based methods struggle.

In **Figure 3**, we observe the convergence of the optimal solution for a three-dimensional Rosenbrock's function ($f(x, y, z) = k(y - x^2)^2 + (1 - x)^2 + k(z - y^2)^2 + (1 - y)^2$) using the MATLAB solver "fminsearch" with k = 100. The convergence behavior shown in **Figure 3** reaffirms the power of using "fminsearch" solver for optimizing Rosenbrock's function. With increased iterations, the solver progressively approaches and finds an improved and accurate optimal solution. This data helps in analyzing the algorithms on various factors and to use those factors to plot graphs of different algorithms.

The comparison **Table 1** also represents the comparison between the computational (CPU) time required for attain the optimal solution of the two-dimensional Rosenbrock's function for various values of k using three different MATLAB solvers (*fminsearch, fminunc, and lsqnonlin*) (15).

TABLE 2 | Numerical comparison of CPUTIME to find optimal solution by MATLAB solver *fmincon* for different values of *k*.

	***fmincon* without gradient**	***fmincon* with gradient**
κ	CPUTIME	CPUTIME
05	1.0296065999	0.7644048999
10	1.0140065000	0.7488048000
20	0.9828062999	0.7176046000
30	0.9672061999	0.7800050000
40	0.9984064000	0.7644049000
50	1.0140065000	0.7332046999
100	0.9828062999	0.7644049000
200	0.9516060999	0.7332046999

Based on the findings presented in **Table 1**, it can be observed that the MATLAB solver "lsqnonlin" exhibits the shortest computational time compared to the other two solvers because it is used for nonlinear least-squares problems where it uses trust-region reflective or Levenberg-Marquardt methods (18, 19). Moreover, this solver demonstrates consistent convergence, which aligns with the earlier discussions.

We employed the MATLAB solver "fmincon" to solve a nonlinear programming problem involving the minimization of a two-dimensional Rosenbrock's function within a specified disk. The problem was approached in two distinct cases:

In Case 1, we provided the gradient information to the solver. In Case 2, we did not provide the gradient information to the solver.

$$\begin{aligned} \min\ & f(x, y) = k(y - x^2)^2 + (1 - x)^2 \\ & \text{subject to } 4x^2 + 9y^2 \leq 2.4 \\ & 9x^2 + 4y^2 \leq 2.4 \\ & x_0 = 0,\ y_0 = 0 \end{aligned} \tag{6}$$

Here, **Table 2** clearly indicates that the computational time taken by the *fmincon* solver is reduced when the gradient information is provided for all cases. **Table 3** presents the results obtained from using the MATLAB solver "fminsearch" to calculate the computational (CPU) time, number of iterations, and optimal solution for Rosenbrock's function across various dimensions (N).

Table 3 clearly demonstrates that as the dimension N increases, both the CPUTIME and the number of iterative steps required to obtain the best solution of Rosenbrock's function also increase. Here it must be mentioned that for $N \leq 5$, the solvers converge efficiently with a good initial guess. For $6 \leq N \leq 7$, the problem becomes more ill-conditioned, and it requires a better initialization and suitable step sizes. Beyond $N > 7$ (not considered here), the problem grows excessively harder due to increased relevancies. As N increases, ill-conditioning magnifies, requiring more adjustable algorithms.

TABLE 3 | Numerical comparison of CPUTIME and number of iterations to find optimal solution by MATLAB solver *fminsearch* for different dimensions.

N	Initial point	CPUTIME	Number of iterations	Optimal solution
2	[0,0]	1.1856076000	79	[1.0000043859, 1.0000106409]
3	[0,0,0]	2.1528137999	165	[0.9999888606, 0.9999784466, 0.9999546148]
4	[0,0,0,0]	3.0108192999	246	[0.9999965740, 0.9999943462, 0.9999873156, 0.9999752880]
5	[0,0,0,0,0]	6.8016435999	550	[0.9999973986, 0.9999915771, 0.9999804183, 0.9999657956, 0.9999319576]
6	[0,0,0,0,0,0]	10.0776645999	795	[0.9954313379, 0.9865170744, 0.9765920213, 0.9572017636, 0.9165731911]
7	[0,0,0,0,0,0,0]	12.3240789999	949	[0.9995159466, 0.9994988146, 0.9993671115, 0.9992656319, 0.9984202982, 0.9968683034, 0.9937796303]

Conclusion

This paper presents a review and comparison of four different optimization toolbox solvers implemented in MATLAB for finding the optimal solution to the nonlinear Rosenbrock's banana function. The solvers evaluated in this study are *lsqnonlin, fminsearch, and fminunc*. The main criteria used for the comparison is the CPUTIME of each solver. Numerical comparisons show that *lsqnonlin* is more efficient than both *fminsearch and fminunc*. It shows faster convergence and less calculation time while reaching the optimal solution for the banana function of Rosenbrock's. Additionally, the paper analyzes the premise of using the MATLAB solver *fminsearch* to identify optimal solutions for approaching multidimensional Rosenbrock's banana functions dimensions ($n = 2$ up to $n = 7$). The solver is demonstrated to be effective in finding the optimal solutions of such higher-dimensional problems. The paper also has a graph with the visualization of the convergence of the optimal solutions for the Rosenbrock's banana functions. The overall effectiveness of lsqnonlin and other MATLAB solvers depends upon various factors such as the class of the optimization problem, smoothness of the function, dimension, and constraints of the function. Its performance may have been good for the Rosenbrock's function, but it may conduct poorly for other problems like highly discontinuous and constrained optimization problems. We recommend that for this reason, the research be extended to include other well-known benchmark functions, such as Ackley, Rastrigin, Griewank (14), and other real-world nonlinear optimization problems. This would serve for further evaluation of the performance of the solver with respect to other MATLAB tools in various situations.

Author contributions

The manuscript's conception, methodology, analysis, writing, and revision were all performed by both authors.

Acknowledgment

Authors acknowledge the institute's authority for encouraging to carry out this research work.

Funding

This research received no specific grant from any funding agency.

References

1. Broyden CG. The convergence of a class of double-rank minimization algorithms 1. general considerations. *IMA J App Math.* (1970) 6:76–90.
2. Drobot S, Tacchi M, Cardozo C, Jones C. SOStab: a Matlab toolbox for transient stability analysis. *Electric Power Syst Res.* (2024) 235: 110812.
3. Kelley CT. *Iterative Methods for Optimization*. Philadelphia: Society for Industrial and Applied Mathematics (1999).
4. Akhtar Z, Rajawat K. Zeroth and first order stochastic Frank-Wolfe algorithms for constrained optimization. *IEEE Trans Signal Process.* (2022) 70:2119–35.
5. Smith H, Norato JA. A MATLAB code for topology optimization using the geometry projection method. *Struct Multidisciplinary Opt.* (2020) 62:1579–94.
6. Coleman TF, Li Y. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM J Opt.* (1996) 6:418–45.
7. Coleman TF, Li Y. On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. *Math Programm.* (1994) 67:189–224.
8. Fletcher R, Powell MJ. A rapidly convergent descent method for minimization. *Comput J.* (1963) 6:163–8.
9. Messac A. *Optimization in Practice with MATLAB®: For Engineering Students and Professionals*. Cambridge: Cambridge University Press (2015).
10. Boyd S, Vandenberghe L. *Convex Optimization*. Cambridge: Cambridge University Press (2004).
11. Thander AK, Paul S, Maitra P. An improved Shamanskii method for finding zeros of linear and nonlinear equations. *Appl Math Sci.* (2012) 6:4277–81.

12. Mandal G, Thander AK, Dey S. Numerical comparison of two different single step iterative methods in complex plane for finding basins of attraction. *Am Institute Phys Conf Ser.* (2023) 2876:040005.
13. Byrd RH, Lu P, Nocedal J, Zhu C. A limited memory algorithm for bound constrained optimization. *SIAM J Sci Comput.* (1995) 16:1190–208.
14. Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, et al. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. *Nat Comput.* (2005) 2005:341–57.
15. MathWorks. "Find Minimum of Unconstrained Multivariable Function – fminunc", MATLAB Documentation. Available online at: https://www.mathworks.com/help/optim/ug/fminunc.html
16. Thander AK, Dasgupta S, Dawn U. A new fourth order newton like iterative method for nonlinear equations. *Appl Math Sci.* (2014) 8:4079–85.
17. Bhattacharyya S, Thander AK. Newton–Krylov subspace method to study structure parameter optimization in rib waveguide communication. *Industry Interactive Innovations in Science, Engineering and Technology: Proceedings of the International Conference, I3SET 2016*. Singapore: Springer (2018). p. 229–38.
18. Thander AK, Bhowmik D. Artificial neural network with the Levenberg-Marquardt algorithm for numerical solution of two-dimension Poisson's equation. *BOHR Int J Smart Comput Inf Technol.* (2023) 4: 55–61.
19. Thander AK, Mandal G. Optical waveguide analysis using alternative direction implicit (ADI) method in combination with successive over-relaxation (SOR) algorithm. *J Opt.* (2024) 53:475–81.
20. Nocedal J, Wright SJ. *Numerical Optimization*. New York, NY: Springer (1999).