

## A Comparison between Genetic Algorithms and Sequential Quadratic Programming in Solving Constrained Optimization Problems

Alaa Sheta, Hamza Turabieh  
*Information Technology Department*  
 Prince Abdullah Bin Ghazi Faculty of Science and Information Technology,  
 Al-Balqa Applied University  
 Al-Salt, Jordan,  
 sheta@bau.edu.jo, H\_turabieh@yahoo.com,  
<http://www.bau.edu.jo>

### Abstract

There are variety of problems in mechanical, electrical, chemical and aerospace engineering that can be formulated as NonLinear Programming (NLPs). The quality of the developed solution significantly affect the performance of such systems. In this paper, we investigate the ability of Genetic Algorithms (GAs) to tackle the constrained NLPs problems. Experimental results indicated that GAs can effectively solve these types of problems. GAs can overcome many problems encountered by traditional search techniques as gradient based methods. The performance of GAs is compared to the Sequential Quadratic Programming (SQP) method.<sup>1</sup>

**Keywords:** *Constraint Optimization, Genetic Algorithms, Quadratic Sequential Programming, Industrial Processes.*

### 1 Introduction

Most industrial manufacturing processes involve dynamic nonlinearity, uncertainty and constraints. Evolutionary Algorithms (EAs) techniques are among those optimization techniques which have been used to solve a variety of optimization problems in industry. Currently there is a growing interest on using EAs to assist providing a reasonable solution for physical nonlinear systems in industry.

EAs include Genetic Algorithms (GAs) [?], Evolutionary Strategies (ESs) [?], Evolutionary Programming [?] and Genetic Programming (GP) [?]. One of the most common applications of GAs is parameter optimization. In [?, ?], Genetic Algorithms and Evo-

lutionary strategies have been used in the parameter identification process of nonlinear systems with various degrees of complexity. In [?, ?] they demonstrate how genetic optimizers can be used to derive superior controller structures in aerospace applications in less time (in terms of function evaluations) than other methods such as Linear Quadratic Regulator (LQR) design. In [?], it was shown that GAs can also be used in the selection and tuning of the controller structure. Genetic Programming was used to model the dynamics of many systems in industry. For example, the winding machine and an automotive engine [?, ?].

For any optimization problem, there is an optimization criterion (i.e. evaluation function) has to be minimized or maximized. The evaluation function represents a measure of the quality of the developed solution. Searching the space of all possible solution is a challenging task. Additional constraints on the domain of search for the parameters makes the problem quite difficult. The constraints might affect the performance of the evolutionary process since some of the produced solutions (i.e individuals) may be unfeasible. Unfeasible solution represents a waste of computation effort.

In fact, it was reported that no general methodology to handle constraints exist although several methods were introduced [?]. Rejecting unfeasible individuals, penalizing unfeasible individuals or moving these individuals to the feasible domain are among the many methods proposed [?, ?].

### 2 Why Genetic Algorithms?

GAs are the most famous among EAs. GAs have been employed as a tool that can handle multi-model function and complex search space. They have the capability to search complex spaces with high probability

<sup>1</sup>This study has been implemented as part of a research grant supported by Al-Balqa Applied University. Al-Salt, Jordan



of success in finding the points of minimum or maximum on the search space (i.e. landscape).

Genetic Algorithms are derivative-free stochastic search algorithms. They apply the concept of natural selection. This idea was first introduced by John Holland at the University of Michigan in 1975 [?]. GAs gain a great popularity due to their known attributes. These attributes include: 1) GAs can handle both continuous and discrete optimization problems; 2) They require no derivative information about the fitness criterion [?, ?]; 3) GAs have the advantageous over other search algorithm since it is less likely to be trapped by local minimum; 4) GAs provide a more optimal and global solution; 5) They are less likely to be trapped by local optimal like Newton or gradient descent methods; 6) In many case studies, they are less sensitive to the presence of noise and uncertainty in measurements [?]; 7) GAs use probabilistic operators (i.e. crossover and mutation) not deterministic ones. For these reasons, they have been successful used in solving numerous applications in engineering and computer science [?, ?].

### 3 Evolutionary Process

The evolutionary process of GAs start by the computation of the fitness of the each individual in the initial population. A flowchart for a simple GA process is given [?] in Figure 1.

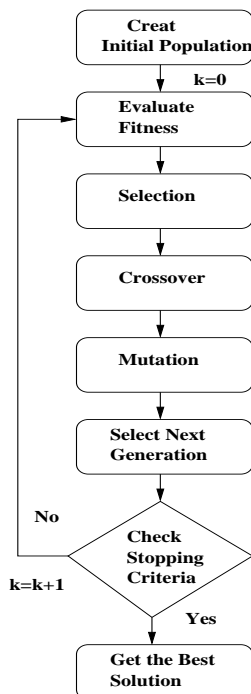


Figure 1: Flowchart of a simple GAs process

To summarize how genetic algorithms work, assume that  $Pop(k)$  and  $Offspring(k)$  are the parents and offspring in current generation  $t$ ; the general

structure of a genetic algorithm procedure can be described by the simple pseudo code.

```

begin
  k=0;
  initialize Pop(k);
  evaluate Pop(k);
  while (termination not reached) do
    recombine Pop(k) to generate Offspring(k);
    evaluate Offspring;
    Select Pop(k + 1) from Pop(k) and Offspring(k);
    k = k + 1
  end while
end
  
```

### 3.1 How GAs Code a Solution?

Genetic algorithms code the candidate solutions of an optimization algorithm as a string of characters which are usually binary digits [?]. In accordance with the terminology that is borrowed from the field of genetics, this bit string is usually called a chromosome (i.e. individuals). A number of chromosomes generate what is called a *population*. The structure for each individual can be represented as follows:

<i>gene<sub>1</sub></i>	<i>gene<sub>2</sub></i>	...	<i>gene<sub>n</sub></i>
11101	00101	...	11011

This a chromosome has number of genes equal to  $n$ . These genes are used in the evaluation function  $f$ . Thus,  $f(gene_1, gene_2, \dots, gene_n)$  is the function to be minimized or maximized.

### 3.2 Selection Mechanism

Selection is the process which guides the evolutionary algorithm to the optimal solution by preferring chromosomes with high fitness. The chromosomes evolve through successive iterations, called *generations*. During each generation, the chromosomes are evaluated, using some measure of fitness. To create the next generation, new chromosomes, called *offspring*, are formulated by using some operators called *crossover* and *mutation*. Thus, a new generation will be created by selecting the best chromosomes (parents) from the previous generation and the best chromosomes from the offspring [?].

### 3.3 Crossover

Crossover is the main genetic operator. In [?] Holland indicates that crossover provides the main search operator while bit mutation simply serves as a background operator to ensure that all possible solutions can enter the population. The probabilities commonly assigned to crossover  $p_c$  and bit mutation  $p_m$  guide the evolutionary process towards the optimal solution. Crossover operates on two chromosomes at a time and



generates offsprings by combining both chromosomes' features. Single-point crossover of two binary string chromosomes can be implemented as follows. Assume we have two parents of chromosomes  $P_1$  and  $P_2$ . A cut-off point selected randomly after bit seven will produce two childes as follows:

$$\begin{aligned} P_1 &= [0011010001] \\ P_2 &= [0001101011] \\ \\ C_1 &= [0011010011] \\ C_2 &= [0001101001] \end{aligned}$$

For other types of representation other crossover types are suggested.

### 3.4 Mutation

Mutation is a background operator which produces some random changes in various chromosomes. In GAs, mutation play the role of replacing the genes lost during the evolutionary process in a new form or produce new genes which were nor explored before in the initial population. One way to do mutation would be to alter one or more genes. Parent, a binary string chromosomes, before  $P_{before}$  and after  $P_{after}$  mutation of a can be presented as follows.

$$\begin{aligned} P_{before} &= [0011010001] \\ P_{after} &= [0001101111] \end{aligned}$$

### 3.5 Fitness Function

GA evaluates the individuals in the population using a selected fitness function (criterion). This function indicates should indicate how good or bad a candidate solution is. The way to select the fitness function is a very important issue in the design of genetic algorithms, since the solution of the optimization problem and the performance of the algorithm count mainly on this function. It is important to recognize that GAs are different from other optimization techniques like gradient descent, since they evaluate a set of solution in the population at each generation and makes them more likely to find the optimum solution [?, ?].

## 4 Nonlinear Programming Problem Formulation

A constrained nonlinear programming problem can be described as follows:

$$\text{Minimize } f(x), \quad x \in F \subseteq S \subseteq R^n \quad (1)$$

Subject to:

$$\begin{aligned} h_i(x) &= 0, \quad i = 1, \dots, p, \\ g_j(x) &\leq 0, \quad j = p + 1, \dots, q, \end{aligned}$$

Given that  $a_k \leq x_k \leq b_k$ ,  $k = 1, \dots, n$ .  $x = [x_1, \dots, x_n]$  is a vector of  $n$  variables.  $f(x)$  is the objective function,  $h_i(x)$  ( $i = 1, \dots, p$ ) is the  $i$ th equality constraint, and  $g_j(x)$  ( $j = p + 1, \dots, q$ ;  $q < n$ ) is the  $j$ th inequality constraint.  $S$  is the whole search space and  $F$  is the feasible search space. The  $a_k$  and  $b_k$  present the lower ad upper bounds of the variable  $x_k$  ( $k = 1, \dots, n$ ), respectively.

## 5 Sequential Quadratic Programming (SQP)

The Sequential Quadratic Programming (SQP) algorithm is a powerful technique for solving nonlinear constrained optimization problems [?]. SQP allows you to closely mimic Newton's method for constrained optimization just as is done for unconstrained optimization. At each iteration, an approximation is made of the Hessian of the Lagrangian function using a quasi-Newton updating method. This is then used to generate a Quadratic subProblem QP subproblem whose solution is used to form a search direction for a line search procedure [?, ?].

In studying the SQP we will adopt the full description of the methodology presented in [?]. SQP is an iterative method which solves at the  $k$  iteration a QP of the following form:

$$\text{Minimize } \frac{1}{2} d^t H_k d + \nabla f(x_k)^t d, \quad (2)$$

Subject to:

$$\begin{aligned} \nabla h_i(x_k)^t d + h_i(x_k) &= 0, \quad i = 1, \dots, p, \\ \nabla g_j(x_k)^t d + g_j(x_k) &\leq 0, \quad i = 1, \dots, p, \end{aligned}$$

$d$  is defined as the search direction and  $h_k$  is a positive definite approximation to the Hessian matrix of Lagrangian function of the problem. The Lagrangian function can be described [?] as:

$$L(x, \gamma, \beta) = f(x) + \sum_{i=1}^p \gamma_i h_i(x) + \sum_{j=p+1}^q \beta_j g_j(x) \quad (3)$$

Where  $\gamma$  and  $\beta$  are the Lagrangian multipliers. The developed quadratic subproblems can then be solved using the active set strategy. The solution  $x_k$  at each iteration is updated according to Equation 4.

$$x_{k+1} = x_k + \alpha_k d_k \quad (4)$$



$\alpha$  is defined as the step size and takes values in the interval  $[0, 1]$ . After each iteration the matrix  $H_k$  is updated based on the Newton Method. One known methods to update the matrix  $H_k$  is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) methods [?]. Thus:

$$H_{k+1} = H_k + \frac{y_k y_k^t}{s_k y_k^t} - \frac{H_k s_k s_k^t H_k}{s_k^t H_k s_k} \quad (5)$$

where:

$$\begin{aligned} s_k &= x_{k+1} - x_k \\ y_k &= \nabla L(x_{k+1}, \gamma_{k+1}, \beta_{k+1}) - \nabla L(x_{k+1}, \gamma_k, \beta_k) \end{aligned}$$

## 6 Constrained Handling Using GAs

Recently, solving constrained optimization problem using genetic algorithms were explored by many authors [?, ?, ?, ?]. In solving optimization problem we normally search for feasible solutions for our problem. During the evolutionary process, many unfeasible solutions appear. This is because some individual in the population due to crossover and mutation operation, might produce solutions outside the feasible search space. Dealing with unfeasible solutions represent a challenge for GAs. Many ways to handle unfeasible solution were proposed [?]. The method we are adopting in this study is fully presented in [?]. This methodology can be described in the following steps:

- The problem constraints can be classified into four types. They are Linear Equalities (LE), Linear Inequalities (LI), Nonlinear Equalities (NE), Nonlinear Inequalities (NI) constraints.
- A random start point is selected for the search. This initial random point should satisfy both LE and LI constraints.
- Set the initial temperature  $\lambda = \lambda_0$ .
- Evaluate each individual in the population using the evaluation function *eval*.

$$eval(X, \lambda) = f(X) + \frac{1}{2\lambda} \sum_{j=1}^m f_j^2(X), \quad (6)$$

- if  $\lambda < \lambda_f$  stop, else
  - decrease  $\lambda$ .
  - use the best individual as an initial solution for the next generation.
  - repeat the previous steps of the algorithms.

This method requires an initial starting temperature  $\lambda_0$  and a final freezing temperature  $\lambda_f$ . A recommended values are, reported in [?],  $\lambda_0 = 1$ ,  $\lambda_{i+1} = 0.1 \times \lambda_i$  with  $\lambda_f = 10^{-6}$ .

Many software packages were used to solve constrained optimization problems written in Optima, MATLAB, GRG and LSGRG. These packages are based on methods such as SQP methods, generalized gradient methods and many others. To develop our results we used two types of programs: 1) The Optimization Toolbox with MATLAB to develop solutions based SQP; 2) The GENOCOP 5.0 software tool which was provided in [?, ?] to solve the constrained optimization problems studied. To run the GENOCOP software we need to specify a set of parameters in an input file. These parameters include the number of variables, the number of equalities, the number of inequalities, the domains specified for each variable. We also specify the population size and the total number of generations. The proposed solution of the constrained optimization problem was compared with the SQP solution [?].

## 7 Test Problem 1

A nonlinear constrained optimization problem described in [?] and extensively discussed in [?, ?] is presented in this section.

$$Min \phi(x, y) = 2x + y \quad (7)$$

Subject to:

$$\begin{aligned} 1.25 - x^2 - y &\leq 0 \\ x + y &\leq 1.6 \end{aligned}$$

Given that  $0 \leq x \leq 1.6$  and  $y \in \{0, 1\}$ . To optimize the above function, we generated the problem surface (i.e. landscape) defined within the given search space. The landscape is shown in Figure 2. To check the performance of the evolutionary process we used various population sizes.

To see how GAs work on various population sizes and to make sure that GAs will converge to the optimal solution. We run GAs with various population sizes as shown in Figure 3. It can be seen that with various population size the optimal value of the function reached the acceptable level.

A comparison between the developed results using the constrained GAs and MATLAB Optimization Toolbox is provided in Table 1. The results show that GAs can provide the same results as the SQP technique. This means that both techniques are effective in this case.



## 8 Test Problem 2

This problem was presented in [?] and was studied in [?].

$$\text{Min } \phi(x_1, x_2, y) = -y + 2x_1 + x_2 \quad (8)$$

Subject to:

$$\begin{aligned} x_1 - 2e^{-x_2} &= 0 \\ -x_1 + x_2 + y &\leq 0 \end{aligned}$$

Given that  $0.5 \leq x_1 \leq 1.4$  and  $y \in \{0, 1\}$ . The above problem can be formulated to eliminate equality constraints as shown in Equation 9.

$$\text{Min } \phi(x_1, y) = -y + 2x_1 - \ln\left(\frac{x_1}{2}\right) \quad (9)$$

Subject to:

$$-x_1 - \ln\left(\frac{x_1}{2}\right) + y \leq 0, y \in \{0, 1\}$$

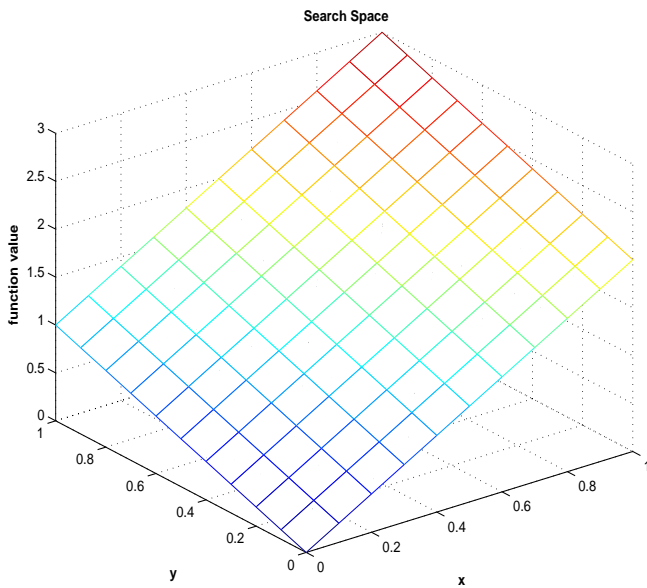


Figure 2: Search space of Test Problem 1

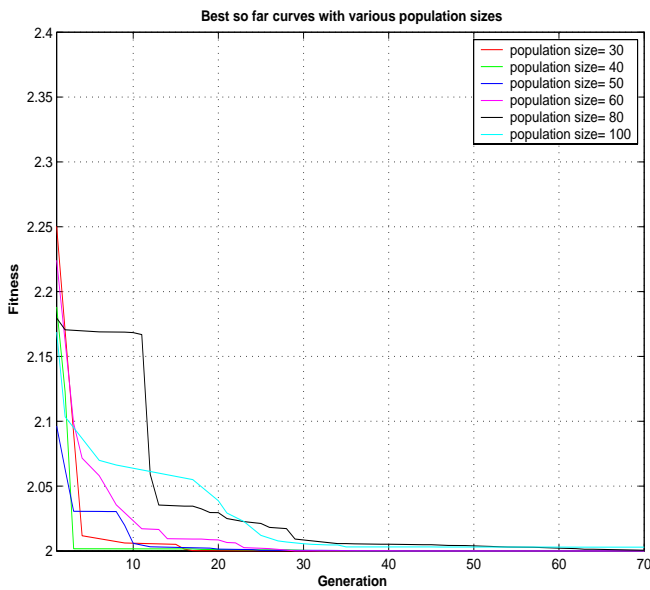


Figure 3: Test Problem 1: Convergence of the evolutionary process with various population sizes

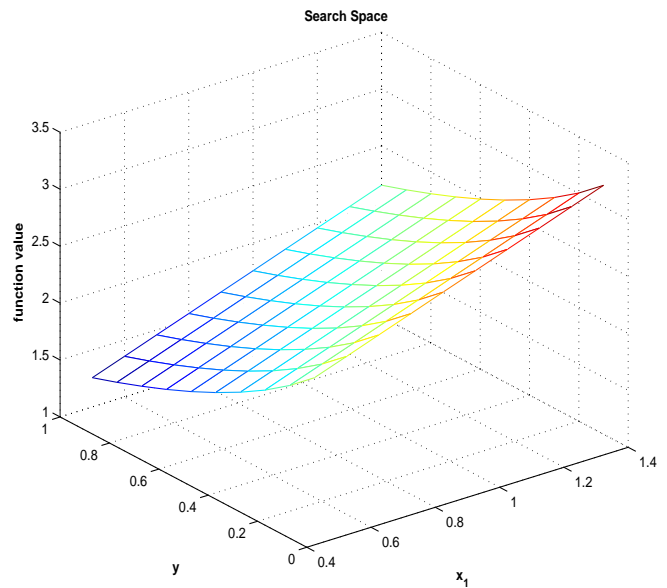


Figure 4: Search space of Test Problem 2

optimal tuning parameters for the second test case using genetic algorithms with constraints. The problem landscape is presented in Figure 4. The landscape seems not very complex but the domain of search space for each model parameters represents a challenge since we are having nonlinear constraints.

In Figure 5, we show the best so far curves of the GAs with various population sizes. In Table 2, GAs provided a slightly better results than the SQP technique.

Table 1: Solution provided by GAs and SQP: Case 1

$x$	$y$	$\phi(x, y)$	Technique
0.5	1	2	GAs
0.5	1	2	SQP



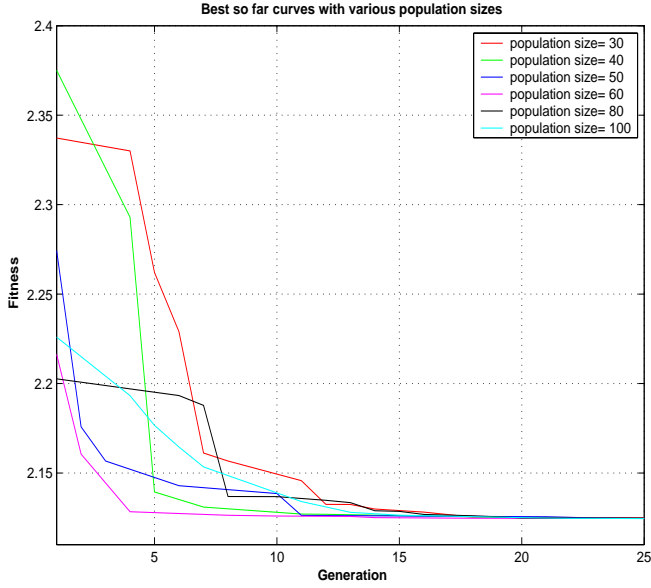


Figure 5: Test Problem 2: Convergence of the evolutionary process with various population sizes

Table 2: Solution provided by GAs and SQP: Case 2

$x_1$	$x_2$	$\phi(x_1, x_2)$	Technique
1.375	1	2.124	GAs
1.3748	1	2.12452	SQP

## 9 Reactor Network Design Problem

The reactor network consist of two Continuous Stirred Tank Reactor (CSTR) where the a sequence of reaction A, B, then C takes place. The design problem objective is to maximize the concentration of product B in the exit stream. This can be achieved by finding the optimal value of the states  $x_1, x_2, x_3, x_4, x_5$  and  $x_6$ . In Figure 6, we show the network design problem structure. The problem under study was fully

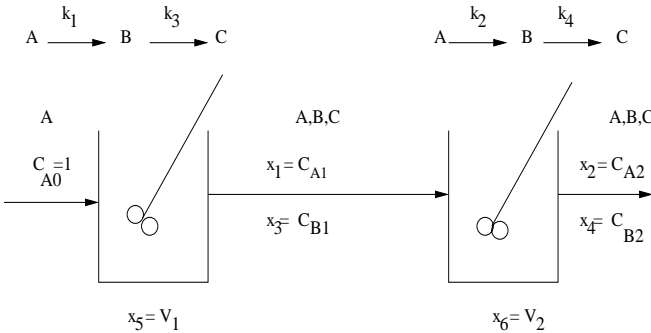


Figure 6: Reactor Network Design Problem

described in [?]. The landscape for the network design problem is shown in Figure 7. The optimization problem can be represented mathematically as given

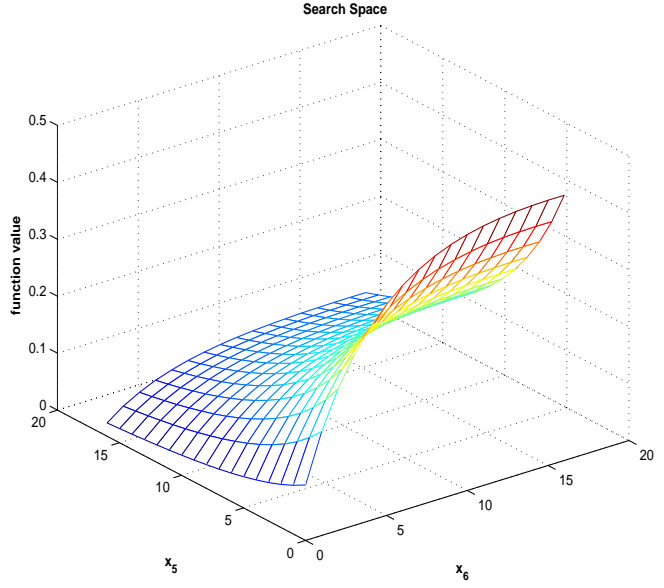


Figure 7: Search space of reactor network design problem

in Equation 10.

$$\text{Min } \phi = -x_4 \tag{10}$$

Subject to:

$$\begin{aligned} x_1 + k_1 x_2 x_5 &= 1 \\ x_2 - x_1 + k_2 x_2 x_6 &= 0 \\ x_3 + x_1 + k_3 x_3 x_5 &= 1 \\ x_4 - x_3 + x_2 - x_1 + k_4 x_4 x_6 &= 0 \\ x_5^{0.5} + x_6^{0.5} &\leq 4 \end{aligned}$$

The domain of search for the states  $x_1, x_2, x_3, x_4, x_5$  and  $x_6$  are given as follows.  $0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1, 0 \leq x_3 \leq 1, 0 \leq x_4 \leq 1, 10^{-5} \leq x_5 \leq 16, 10^{-5} \leq x_6 \leq 16$ . The values of the coefficient  $k_1, k_2, k_3$  and  $k_4$  are given as:

$$\begin{aligned} k_1 &= 0.09755988 \\ k_2 &= 0.99k_1 \\ k_3 &= 0.0391908 \\ k_4 &= 0.9k_3 \end{aligned}$$

To deal with the above problem, we decide to transfer the problem to a maximization problem by eliminating the equality constraints. The new mathematical description can be given as in Equation 11. with the boundary values of  $x_5$  and  $x_6$  are  $10^{-5} \leq x_5 \leq 16, 10^{-5} \leq x_6 \leq 16$ .

$$\text{Max } \phi = \frac{k_2 x_6 (1 + k_3) + k_1 (1 + k_2 x_6)}{(1 + k_1 x_5)(1 + k_2 x_6)(1 + k_3 x_5)(1 + k_4 x_6)} \tag{11}$$



Table 3: Solution provided by GAs and SQP: Reactor Network Problem

$x_5$	$x_6$	$\phi(x_5, x_6)$	Technique
3.038	5.096	0.3881	GAs
15.975	1e-005	0.3746	SQP

Subject to:

$$x_5^{0.5} + x_6^{0.5} \leq 4$$

To maximize the function  $\phi(x_5, x_6)$ , we used both GAs and SQP. We ran GA with population sizes 30, 40, 50, 60, 80 and 100 and computed the best solution after each generation. The results of each run are shown in Figure 8. In this case, GAs outperform

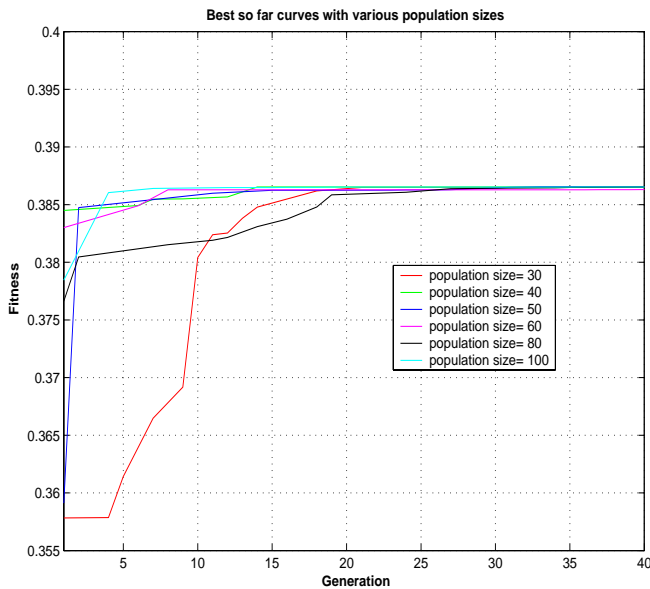


Figure 8: Reactor Network Problem: Convergence of the evolutionary process with various population sizes

SQP in providing a better maximum to the function  $\phi(x_5, x_6)$ . The results for each case are shown in Table 3.

## 10 Conclusion

In this paper, we used Genetic Algorithms (GAs) to solve constrained optimization problems for a number of processes. We explored the performance of the evolutionary process under variations in the population size. The results show that GAs are robust and can provide optimal solution after each run. A practical example of an industrial process, the reactor network design problem, was studied with promising results. A comparison between SQP and GAs was provided. It shows that GAs can outperform SQP in solving complex constrained optimization problems for industrial processes.

## Acknowledgements

Authors would like to acknowledge the financial support of Al-Balqa Applied University, Al-Salt, Jordan under Grant No.: 62/2005.

## References

- [1] H. Al-Duwaish and W. Naeem. Nonlinear model predictive control of hammerstein and winner model using genetic algorithms. In *Proceedings of the IEEE International Conference on Control Applications*, pages 465–469, 2001.
- [2] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–24, 1993.
- [3] M. Biggs. Constrained minimization using recursive quadratic programming. In *Towards Global Optimization*, pages 341–349. North-Holland, 1975.
- [4] M.F. Bramlette and R. Cusin. A comparative evaluation of search methods applied to parameter design of aircraft. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 213–218, 1989.
- [5] J. Claverie, K. De Jong, and A. Sheta. Robust nonlinear control design using competitive coevolution. In *Proceedings of the Congress on Evolutionary Computation (CEC2000)*, pages 403–409, 2000.
- [6] K. A. De Jong. *Analysis of Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Arbor, MI, 1975.
- [7] R. Fletcher. *Practical Methods of Optimization*. A Wiley-Interscience Publications, John Wiley & Sons, Chichester, 1987.
- [8] C. A. Floudas, A. Aggarwal, and A. R. Ciric. Global optimum search for nonconvex NLP and MINLPs problems. *Computers and Chemical Engineering*, 13:1117–1132, 1989.
- [9] C. J. Fogal, A. J. Owens, and M. J. Walsh. *Artificial Intelligence Through Simulated Evolution*. Wiley Publishing, New York, 1966.
- [10] M. Gen and R. Cheng. *Genetic Algorithms and Engineering Design*. Jonh Wiley and Sons, Inc, New York, 1997.
- [11] D. E. Goldberg. *Computer-Aided Gas Pipeline Operation Using Genetic Algorithms and Rule Learning*. PhD thesis, University of Michigan, Ann Arbor, 1983.



- [12] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, New York, 1989.
- [13] S. Han. A globally convergent method for nonlinear programming. In *Journal Optimization Theory and Applications*, volume 22, page 279, 1977.
- [14] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.
- [15] A. Hussian, A. Sheta, M. Kamel, M. Telbany, and A. Abdelwahad. Modeling of a winding machine using genetic programming. In *Proceedings of the Congress on Evolutionary Computation (CEC2000)*, pages 398–402, 2000.
- [16] G. R. Kocis and I. E. Grossmann. Relaxation strategy for the structural optimization of process flow sheets. *Industral and Engineering Chemistry Research*, 26:1869–1880, 1987.
- [17] G. R. Kocis and I. E. Grossmann. Global optimization of nonconvex mixed-integer nonlinear programming MINLP problems in process synthesis. *Industral and Engineering Chemistry Research*, 27:1407–1421, 1988.
- [18] J. R. Koza. *Genetic Programming II*. MIT Press, 1994.
- [19] S. Koziel and Z. Michalewicz. A decoder-based evolutionary algorithm for constrained parameter optimization problems. In *Proceedings of the 5th Parallel Problem Solving from Nature*, pages 231–240. Springer-Verlag, Lecture Notes in Computer Science, Lecture Notes in Computer Science, A.E. Eiben, T. Baeck, M. Schoenauer, and H.-P. Schwefel (Editors), 1998.
- [20] S. Koziel and Z. Michalewicz. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation*, 7(1):19–44, 1999.
- [21] K. Krishnakumar and D. E. Goldberg. Control system optimization using genetic algorithms. *Journal of Guidance, Control and Dynamics*, 15(3):735–740, 1992.
- [22] Z. Michalewicz and N. Attia. Evolutionary optimization of constrained problems. In *Proceedings of the Third Annual Conference on Evolutionary Programming*, pages 98–108. eds. A.V. Sebald and L. J. Fogel, River Edge, NJ, World Scientific Publishing, 1994.
- [23] Z. Michalewicz and C. Janikow. Handling constraints in genetic algorithms. In *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 151–157. Los Altos, CA, Morgan Kaufmann Publishers, 1991.
- [24] Z. Michalewicz, T. D. Logan, and Swaminathan. Evolutionary operators for continuous convex parameter spaces. In *Proceedings of the Third Annual Conference on Evolutionary Programming*, pages 84–97. eds. A.V. Sebald and L. J. Fogel, River Edge, NJ, World Scientific Publishing, 1994.
- [25] M.J.D. Powell. The convergence of variable metric methods for nonlinearly constrained optimization calculations. In *Nonlinear Programming, (O.L. Mangasarian, R.R. Meyer and S.M. Robinson, eds.)*, Academic Press, volume 3, 1978.
- [26] M.J.D. Powell. A fast algorithm for nonlinearly constrained optimization calculations. In *Numerical Analysis, G.A. Watson ed., Lecture Notes in Mathematics, Springer Verlag*, volume 630, 1978.
- [27] J.T. Richardson, M.R. Palmer, G. Liepins, and M. Hilliard. Some guidance for genetic algorithms with penalty functions. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 191–197. Morgan Kaufmann Publisher, Los Altos, CA, 1989.
- [28] H. S. Ryoo and B. P. Sahinidis. Global optimization of nonconvex NLPs and MINLPs with application in process design. *Computers and Chemical Engineering*, 19:551, 1995.
- [29] R. L. Salcedo. Solving nonconvex nonlinear programming problems with adaptive random search. *Industral and Engineering Chemistry Research*, 31:262, 1992.
- [30] A. Sheta and A. H. Abdelwahab. Identification of nonlinear communication channel using evolutionary volterra time-series. In *Proceedings of the Congress on Evolutionary Computation, Washington, DC, July*, pages 229–235, 1999.
- [31] A. Sheta and J. Gertler. Modeling the dynamics of an automotive engine using genetic programming. In *Proceedings of the International Symposium on Engineering of Natural and Artificial Intelligent Systems (ENAIIS2001)*, American University in Dubai, U.A.E., 2000.
- [32] A. Sheta and K. De Jong. Parameter estimation of nonlinear systems in noisy environment using genetic algorithms. In *Proceedings of the IEEE International Symposium on Intelligent Control (ISIC'96)*, pages 360–366, 1996.
- [33] A. Varsek, T. Urbancic, and B. Filipic. Genetic algorithms in controller design and tuning. *IEEE Transaction Syst. Man. Cybern.*, 23(5), 1993.
- [34] Ozgur Yeniay. A comparative study on optimization methods for the constrained nonlinear programming problems. *Mathematical Problems in Engineering*, 2:165–173, 2005.

