EVOLUTIONARY COMPUTATION STRATEGY FOR OPTIMIZATION OF AN ALKYLATION REACTION

B.V.Babu* **& Gaurav Chaturvedi**
Department of Chemical Engineering
Birla Institute of Technology and Science
Pilani-333031 (Rajasthan) India

Abstract

The alkylation process is an important reaction used in refineries to upgrade light olefins and isobutene into a much more highly valued gasoline component. The primary alkylation reaction involves the reaction of isobutene with a light olefin such as butylenes, in the presence of strong acid catalyst to form the high octane. A problem of long standing is to determine the optimal operating conditions for the alkylation process. In order to achieve this, Differential Evolution has been employed. An improved scheme of Genetic Algorithms, Differential Evolution is exceptionally simple, fast and robust. It is primarily a search algorithm that is able to locate near optimal solutions to difficult problems. An analysis of the results obtained on varying the parameters of the problem is also presented.

Key Words: Alkylation process, Optimization, Evolutionary Computation, Differential Evolution, Genetic Algorithms

1. Introduction

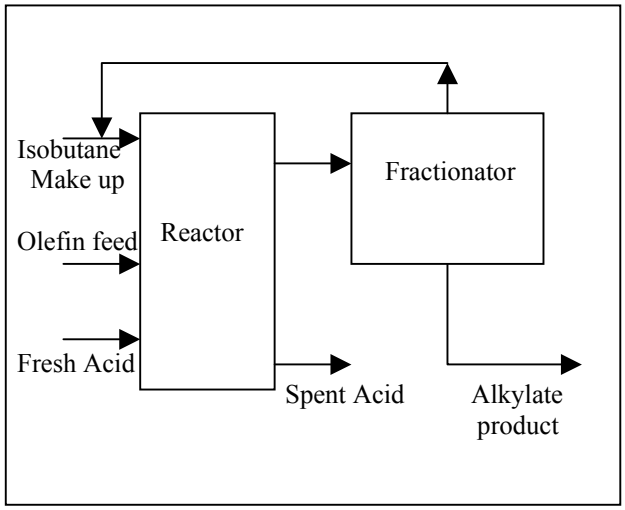
The alkylation process is an important unit that is used in refineries to upgrade light olefins and isobutane into much more highly valued gasoline component. The light olefins are produced mainly from catalytic crackers and also from cokers and vis breakers.

The primary alkylation reaction involves the reaction of isobutane with a light olefin, such as butylene, in the presence of strong acid catalyst to form the high octane, trimethyl pentane isomer. The alkylate obtained is one of the best gasoline blending components produced in the refinery because of its high octane, typically 96RON, and low vapour pressure which allows lower cost butane to be put into gasoline. The high profitability of upgrading light olefins and isobutane in the refinery from LPG value to gasoline value explains why it is a very popular process alternative that is used at most locations that have catalytic crackers.

Alkylation technology has been used for a long time in the refining industry. A tank reactor system was first installed in Exxon's Baytown refinery in the late 1930's.

A problem of long standing is to determine the optimal operating conditions for the simplified alkylation process shown in Fig. 1 Sauer et al. (1964) solved this problem using a form of successive linear programming. The problem was also attempted by Himmelblau et al. (1993) by successive quadratic programming. They utilized the computer code NPSOL to solve the problem. In the formulation provided by them, the objective function was defined in terms of the alkylate product, or output value minus feed and recycle costs, operating costs were not reflected in the function. The notation used is shown in Table-1 along with the upper and lower bounds on each variable. The bounds represent economic, physical and performance constraints.

Figure 1.: A simple alkylation reaction



* Corresponding Author: Group Leader, ET Group, ESD (Workshop), BITS, Pilani; E-mail: bvbabu@bits-pilani.ac.in; Fax: (01596)44183; Phones: (01596)45073 Ext. 205(Off.); (01596) 42212/44977 (Res.).

Table-1: Variables and Bounds.

Symbol	Variable	Lower Bound	Upper Bound
x_1	Olefin feed(barrels/Day)	0	2000
x_2	Isobutane recycle(barrels/day)	0	16000
x_3	Acid Addition Rate (X1000 pounds/day)	0	120
x_4	Alkylate yield (barrels/ day)	0	5000
x_5	Isobutane makeup (barrels/day)	0	2000
x_6	Acid strength (wt. %)	85	93
x_7	Motor octane no.	90	95
x_8	External Isobutane-olefin Ratio	3	12
x_9	Acid dilution factor	1.2	4
x_{10}	F-4 performance no.	145	162

The total profit per day to be maximized is given by the following function

$$f(x) = C_1 x_4 x_7 - C_2 x_1 - C_3 x_2 - C_4 x_3 - C_5 x_5 \quad \text{……..(a)}$$

where C_1 = Alkylate product value($0.063 per octane barrel)
C_2 = Olefin feed cost ($5.04 per barrel).
C_3 = isobutane recycle costs ($0.035 per barrel).
C_4 = acid addition costs ($10 per 1000 pounds).
C_5 = isobutane make up cost ($3.36 per barrel).

Himmelblau et al, carried out regression analysis to form the process model. The alkylate yield, x_4, was a function of the olefin feed, x_1, and the external isobutane- to- olefin ratio, x_8. The relationship determined by holding the reactor temperature between 80^o F and 90^o F and the reactor acid strength by weight percent at 85 to 93 was:

$$x_4 = x_1(1.12 + 0.13167x_8 - 0.0067x_8^2) \quad \text{……...(b)}$$

The isobutane makeup, x_5, was determined by a volumetric reactor balance. The alkylate yield, x_4, equals the olefin feed, x_1, plus the isobutane make up, x_5, less shrinkage. The volumetric shrinkage can be expressed as 0.22 volume percent of alkylate yield so that

$$x_4 = x_1 + x_5 - 0.22x_4$$

or

$$x_5 = 1.22x_4 - x_1 \quad \text{…….(c)}$$

The acid strength by weight percent, x_6, could be derived from an equation that expressed the acid addition rate, x_3, as a function of the alkylate yield, x_4, and the acid dilution factor, x_9, and the acid strength by weight percent, x_6. (the addition acid was assumed to have a strength of 98%).

$$1000x_3 = \frac{x_3 . x_9 . x_6}{98 - x_6}$$

or

$$x_6 = \frac{98000x_3}{x_4 . x_9 + 1000x_3} \quad \text{……....(d)}$$

The motor octane number, x_7, was a function of the external isobutane- to- olefin ratio, x_8, and the acid strength by weight percent, x_6, (for the same reactor temperatures and acid strengths as for the alkylate yield, x_4).

$$x_7 = 86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89) \quad \text{……...(e)}$$

The external isobutane- to- olefin ratio, x_8, was equal to the sum of the isobutane recycle, x_2, and the isobutane make up, x_5, divided by the olefin feed, x_1.

$$x_8 = \frac{x_2 + x_5}{x_1} \quad \text{……...(f)}$$

The acid dilution factor could be expressed as a linear function of the motor octane number, x_9, and the F 4 performance number, x_{10}.

$$x_9 = 35.82 - 0.222x_{10} \quad \text{…….(g)}$$

The last dependent variable is x_{10}, which was expressed as a linear function of the motor octane, x_7.

$$x_{10} = -133 + x_7 \quad \text{…….(h)}$$

Equations (c), (d) and (f) were used as equality constraints. The other relations were modified to form two inequality constraints each so as to take into account the uncertainty that existed in their formulation. The d_l (lower) and d_u (upper), listed in Table-2, allow for deviations from expected values of the associated variables.

Table-2: Deviation Parameters

Deviation Parameter	Value
d_{4l}	0.99
d_{4u}	100/99
d_{7l}	0.99
d_{7u}	100/99
d_{9l}	0.9
d_{9u}	10/9
d_{10l}	0.99
d_{10u}	100/99

Thus there were eight inequality constraints in addition to the three equality constraints and the upper and lower bounds on all of the variables, the details of which are shown in Eqs. (i) – (n) below:

$$[x_1(1.12 + 0.13167x_8 - 0.0067x_8^2)] - d_{4l}x_4 \geq 0 \quad \text{……..(i)}$$

$$-[x_1(1.12 + 0.13167x_8 - 0.0067x_8^2)] - d_{4u}x_4 \geq 0 \quad \text{……..(j)}$$

$$[86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_6 - 89)] - d_{7l}x_7 \geq 0 \quad \text{……..(k)}$$

$$-[86.35 + 1.098x_8 - 0.038x_8^2 + 0.325(x_8 - 89)] + d_{7u}x_7 \geq 0 \quad \text{……..(l)}$$

$$[35.82 - 0.222x_{10}] - d_{9l}x_9 \geq 0 \quad \text{…….(m)}$$

$$-[35.82 - 0.222x_{10}] - d_{9u}x_9 \geq 0 \quad \text{……..(n)}$$

$$[-133 + 3x_7] - d_{10l}x_{10} \geq 0 \quad \text{……..(o)}$$

$$-[133 + 3x_7] + d_{10u}x_{10} \geq 0 \quad \text{……..(p)}$$

The optimal values obtained by the code NPSOL (Himmelblau et al., 1993) are listed in Table-3 along with the maximized objective function value ($f(x^*)$).

Table-3:Optimal Values

Variable	Optimal Value
x_1	1698.1
x_2	15819
x_3	54.107
x_4	3131.2
x_5	2000
x_6	90.115
x_7	95
x_8	10.49
x_9	1.56
x_{10}	153.54

2. **Differential Evolution**

Since their inception three decades ago, Genetic Algorithms (GA) have evolved like the species they try to mimic (Goldberg, 1989). Just as competition drives species to adapt a particular environmental niche, so too, has the pressure to find efficient solutions across the spectrum of real world problems forced Genetic Algorithms to diversify and specialize (Androulakis & Venkatasubramanian, 1991; Stair & Fraga, 1995; Upreti & Deb, 1996; Babu & Vivek, 1999; Babu & Mohiddin, 1999).

Differential Evolution is a search procedure similar to GA applied on real world variables that is significantly fast at numerical optimization and is also more likely to find a function's true global optimum (Price and Storn, 1997). Among DE's advantages are its simple structure, ease of use, speed and robustness. DE has been used to design several complex digital filters and to design fuzzy logic controllers, for estimation of parameters and for design and control applications (Storn, 1995; Masters & Land, 1997; Wang et al. 1998; Chiou and Wang, 1999; Joshi & Sauderson, 1999; Babu & Sastry, 1999; Babu & Munawar, 2000a & 2000b; Babu & Rishindra Pal Singh, 2000).

The inherent advantages of DE over GA emerge from that it uses floating point numbers instead of bit strings for coding and uses addition for mutation. The former does not limit the resolution with which an optimum can be located with precision as is the case with bit strings that can code integers. Wolf and Moros (1997) used floating point code but that incorporates additional complexity. To see the advantages of addition for mutation instead of bit flipping consider the binary 15 (01111). To change it to binary 16 (10000) would involve flipping of all the bits, though adjacent transitions in mutation should be the frequent. DE uses the population itself to select how much to add, thus

automatically scaling the magnitude and preventing uphill moves.

The overall structure of DE proposed by Price and Storn resembles that of other population based searches. A population of NP vectors is first initialized between some realistic upper and lower bounds, if available, otherwise random initialization is resorted to. In each generation, NP competitions are held to determine the composition of the next generation. In particular, the i^{th} competition puts the i^{th} vector, known as the target vector against its adversary, the trial vector. The trial vector's parents are interestingly the target vector itself and a randomly chosen population vector to which a weighted random difference vector has been added. Mating is non- uniform and is controlled by a crossover constant(CR).

The other parent can be represented as:

$$X_c' = X_c + F(X_a . X_b)$$

where X_c , X_a , X_b are randomly chosen vectors.

Which parent contributes which parameters are decided by a series of binomial experiments. A random number in [0,1] is generated, if it is greater than CR, that trait is inherited from the target vector otherwise it comes from the random vector X_c' . To ensure that the trial vector differs from the target by at least one parameter, the final vector parameter comes form X_c' even when CR=0.

DE's selection criteria resemble tournament selection in that the target and the trial vector are pitted in a winner take all competition. The one with the lower cost advances to the next generation.

3. Results and Discussion

The pseudo code for implementing DE is given below:

- Initialize variables of maximum population, F,CR and maximum number of generations.
- Define structure of an individual solution with a single dimensional array of 10 variables (x_1 to x_{10}) and with fitness of solution.

Till the population size is less than 100, iterate.

- Initialize the individual members of the population with randomly chosen values between the upper and lower limits. Compute the values of the dependent variables (x_2, x_5, x_6) using the independent members.

Till the maximum generations are not reached or the solution does not converge, iterate.

- For I=0, I<population size(NP), find three random array positions different from the running index and mutate.
- Cross the individual obtained after mutation with the parent.
- If the fitness of parent is more than the offspring then the parent goes to the next generation and vice versa.

Terminate the two loops.

For the implementation of DE with a constrained problem, solutions were penalized on the fitness if any of the constraints were violated. The magnitude of the penalty was adjusted so as to significantly affect the fitness.

The solution was attempted with different combinations of values F,CR, population sizes from 5 to 100 and for a maximum of 1000 generations. The solutions obtained were evaluated on two criteria- quality of the solution (by measuring the standard deviation(SD) of the solution from that obtained of NPSOL. SD was defined as:

$$SD = \sqrt{\sum_{i=1}^{10} \frac{(x_i - x_{ia})^2}{n-1}}$$

where,

x_i = i^{th} parameter obtained by DE

x_{ia} = i^{th} parameter obtained by NPSOL

n = 10) and the ease of convergence (by measuring the number of generations it converges in. Convergence was deemed to have been obtained if the objective function value changed by less than 0.1% over 100 generations)

The trends in the results based on the ease of convergence are:

- For high values of CR depending on F, solutions with low population sizes do not converge. The values of CR beyond which significant non-convergence is observed for low population sizes are CR=0.5 for F=0.4, CR=0.3 for F=0.5, CR=0.5 for F=0.6, CR=0.6 for F=0.7. Beyond F=0.7, non-convergence is observed for all population sizes and hence are not preferable.
- For low values of CR again depending on F, solutions with high population sizes do not converge. Again for high values of F (>0.6) the non- convergence with low CR is not restricted to high population sizes and substantial non-convergence is observed for all population sizes.
- Thus, values of F<0.6 are optimal for this problem. As is seen from Figs. 1-5, value of F=0.4 and F=0.5 converge for widest range of CR values(terminating lines show that no convergence was obtained beyond that value of CR) for all population sizes.

Another way of finding ease of convergence is to find the number of generations it took for the solution to converge. The following trends were visible:

- On increasing population size, the number of iterations increase slightly for all combinations of CR and F. In fact, for a given combination of CR, number of generations are mostly within ± 200 of each other for all population sizes.
- For a given F, the number of iterations required for convergence increase slightly with decreasing CR. This is true for all F's except high F's(≥ 0.7) where convergence is too infrequent to speak of any trend.
- For a given CR, with increasing F, number of generations required generally increases.

For maximum speed, we should thus have a low F and a high CR(F=0.4, CR≥0.7) with low population size.

To use the other criterion i.e., quality of solution, a rather different picture is painted. The SD of the solution increases for all F's as CR increases beyond 0.6. That is, it is not much greater than the SD for lower values of CR. For values of F ≥ 0.7, there is no trend because of rare convergence. As the population size increases the solutions usually have a lower SD and are more accurate.

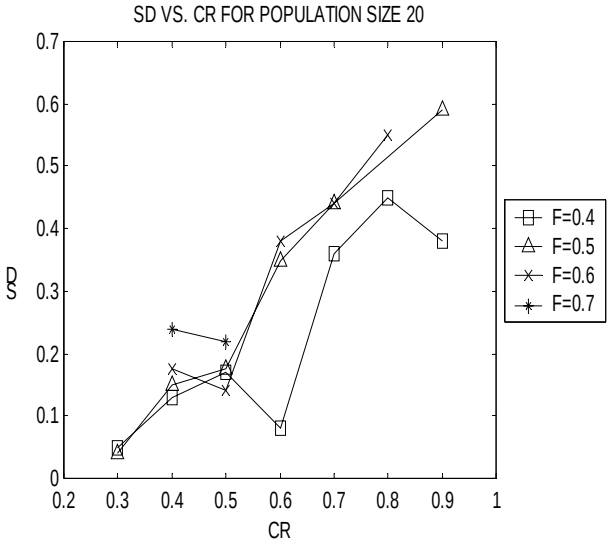


Figure. 2

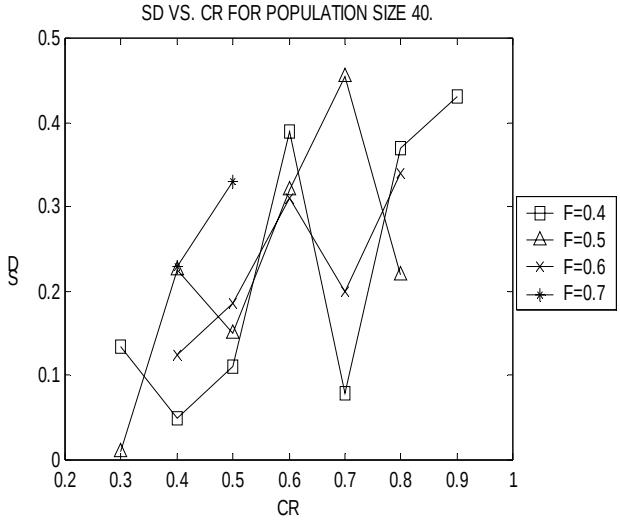


Figure. 3

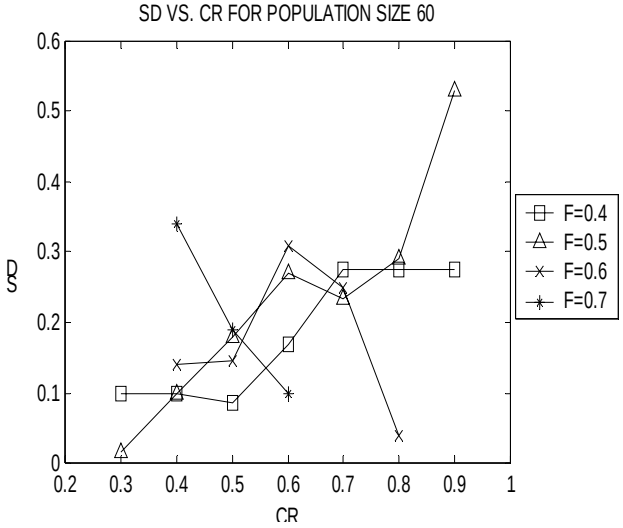


Figure. 4

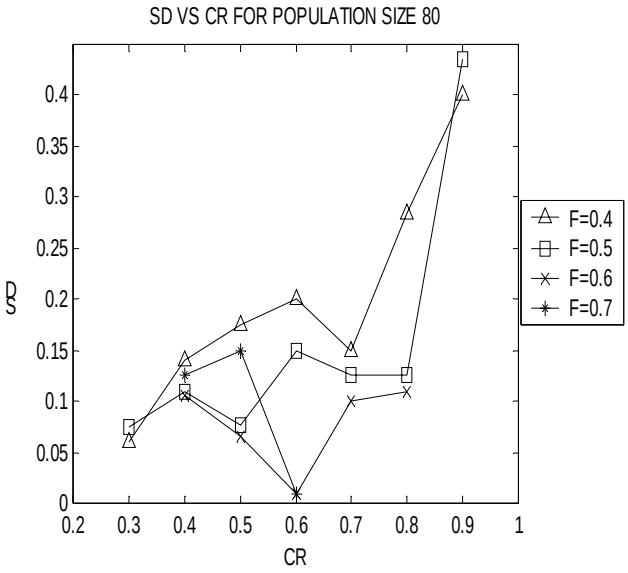


Figure. 5

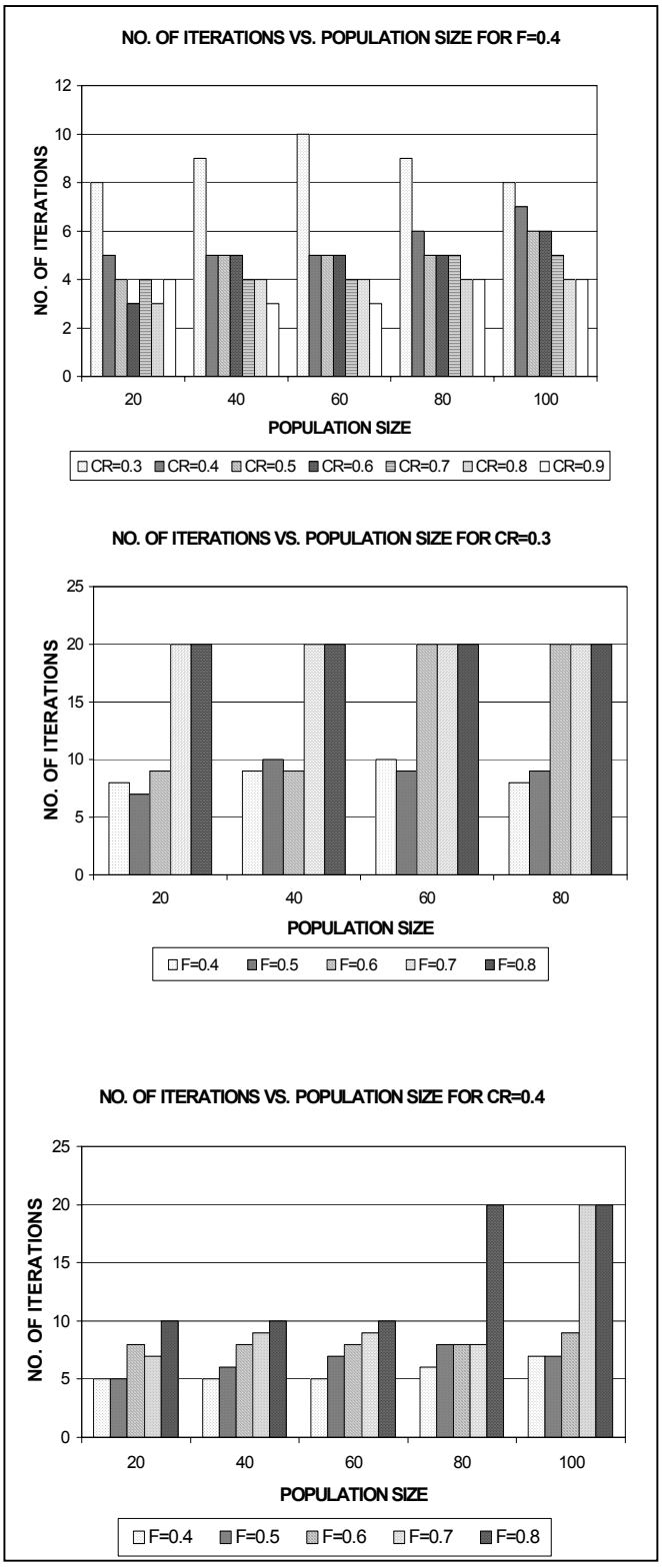


Figure. 7

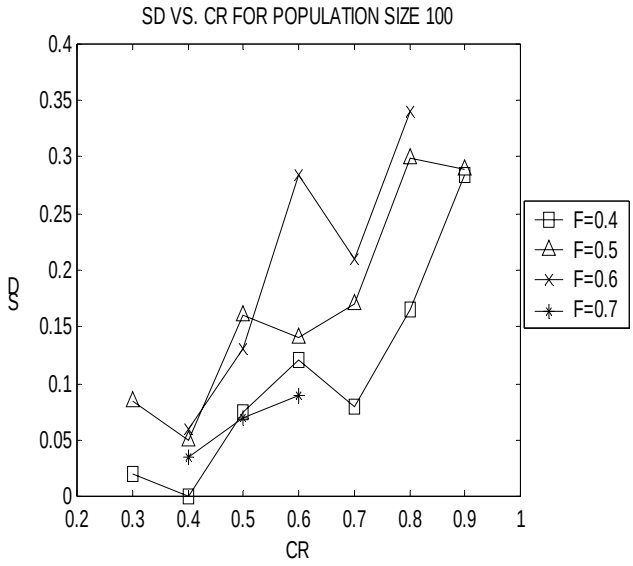
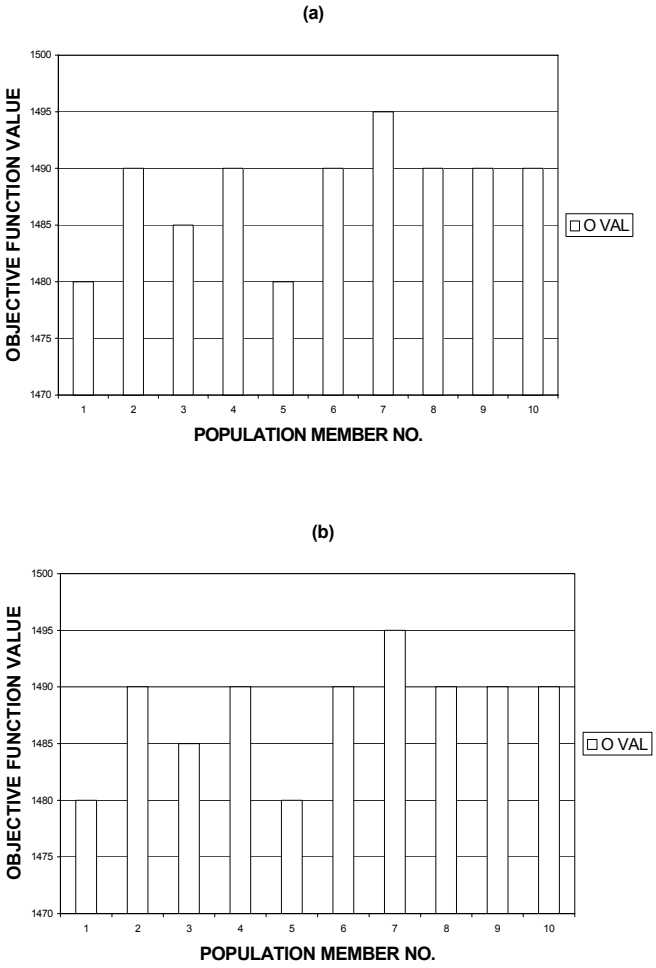
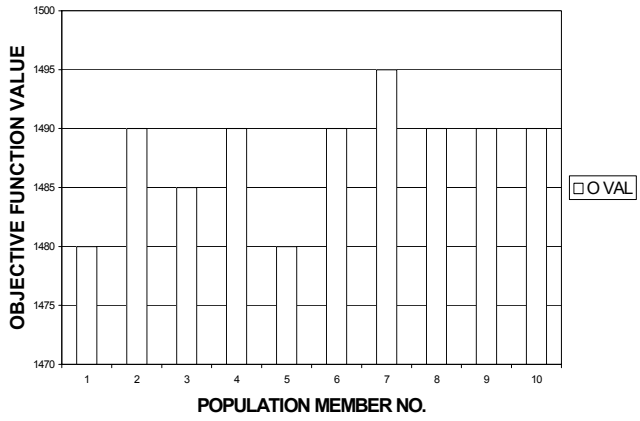


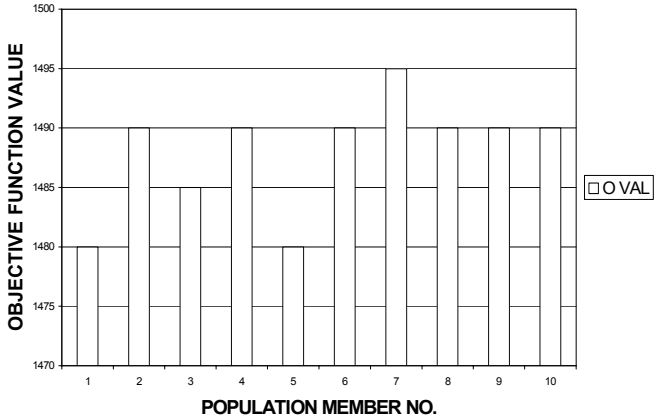
Figure. 6



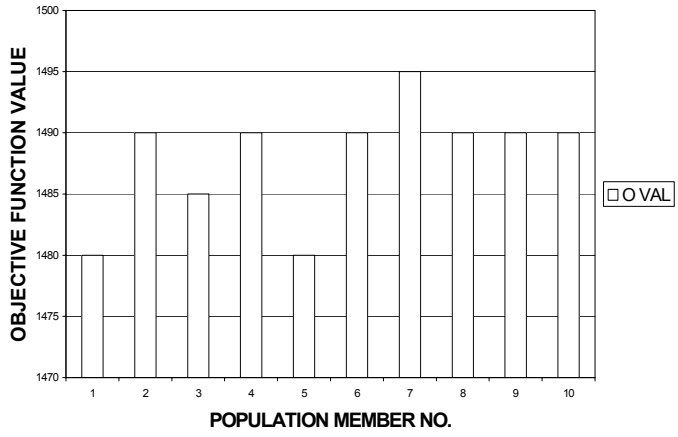
(c)



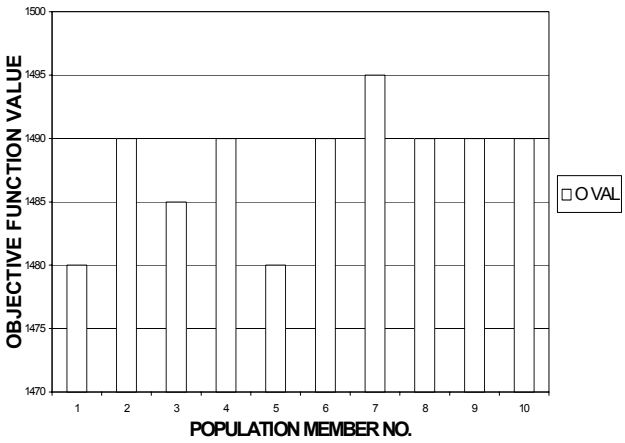
(f)



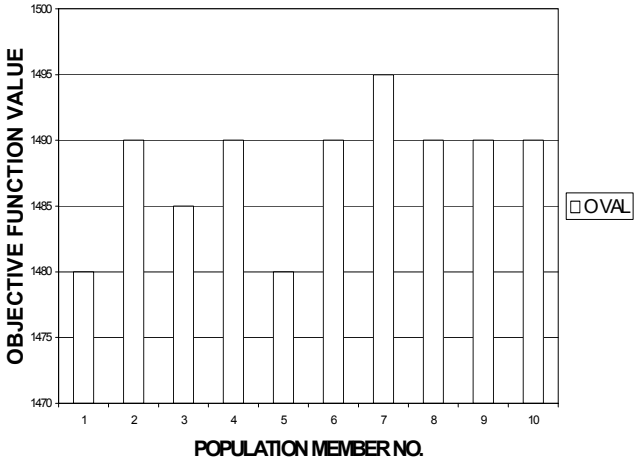
(d)



(g)



(e)



Figures. 8(a)-8(g): Objective Function values for member nos. for various generations

4. Conclusion

The analysis depicts that there are trade off's associated with the parameters. Large population sizes give a more exact solution but take more time. High values of CR are also conducive to convergence but are again slower. Low F values are favorable on both counts.

The best solution possible for the given problem appears to be with a value of F=0.4, moderately high CR=0.6 and a population size of 20, that converges in only 300 iterations and has an SD less than 0.07.

5. References

Androulakis, I. P. & Venkatasubramanian, V. (1991). A Genetic Algorithm framework for process design and optimization. *Computers and Chemical Engineering, 15*(4), 217- 228.

Babu, B. V. & Mohiddin, S. B. (1999). Automated Design of Heat Exchangers Using Artificial Intelligence based optimization. *Proceedings of 52nd Annual Session of IIChE (CHEMCON- 99),* Chandigarh, India, December 20- 23, 1999.

Babu, B. V. & Munawar, S. A. (2000a). Differential Evolution for the optimal design of heat exchangers. *Proceedings of All- India seminar on Chemical Engineering Progress on Resource Development: A Vision 2010 and Beyond,* IE (I), Bhubhaneswar, India, March 11, 2000.

Babu, B. V. & Munawar, S. A. (2000b). Optimal Design of Shell & Tube Heat Exchanger by Different Strategies of Differential Evolution. *Communicated to Computers and Chemical Engineering.*

Babu, B. V. & Rishindra Pal Singh (2000). Synthesis & Optimization of Heat- Integrated Distillation Systems using Differential Evolution. *Proceedings of All- India Seminar on Chemical Engineering Progress on Resource Development: A Vision 2010 and Beyond,* IE (I), Bhubhaneswar, India, March 11, 2000.

Babu, B. V. & Sastry, K. K. N. (1999). Estimation of heat transfer parameters in a trickle bed reactor using differential evolution and orthogonal collocation. *Computers and Chemical Engineering, 23*, 327- 339.

Babu, B. V. & Vivek, N. (1999). Genetic algorithms for estimating heat transfer parameters in trickle bed reactors. *Proceedings of 52nd Annual Session of IIChE (CHEMCON- 99),* Chandigarh, India, December 20- 23, 1999.

Chiou, J. P. & Wang, F. S. (1999). Hybrid method of evolutionary algorithms for static and dynamic optimization problems with application to a fed-batch fermentation process. *Computers & Chemical Engineering, 23*(9), 1277- 1291.

Goldberg, D. E. (1989). *Genetic Algorithms in search, optimization, and machine learning,* Reading, MA: Addison- Wesley.

Joshi, R. & Sanderson, A. C. (1999). Minimal representation multi- sensor fusion using Differential Evolution. *IEEE Transactions on Systems, Man and Cybernetics, Part A* 29(1), 63-76.

Masters, T. & Land, W. (1997). A new training algorithm for the general regression neural network. 1997 *IEEE International conference on Systems, Man and Cybernetics, Computational cybernetics and Simulation, 3,* 1990- 1994.

Mcleod, A. S., Johnston, M. E. & Gladden, L. F. (1997). Development of a genetic algorithm for a molecular scale catalyst design. *Journal of Catalysis, 167,* 279- 285.

Moriyama, H. & Shimizu, K. (1996). On- line optimization of culture temperature for ethanol fermentation using a genetic algorithm. *Journal of Chemical Technology and Biotechnology, 66,* 217- 222.

Price, K. & Storn, R. (1997). Differential Evolution. *Dr. Dobb's Journal.* 18-24.

Sauer, R. N., A. R. Corville & C. W. Burwick (1964). Computer Points Way to More Profits. *Hydrocarbon Processing. 63.* (June, 1984).

Stair, C. & Fraga, E. S. (1995). Optimization of Unit Operating Conditions for heat- integrated processes using genetic algorithms. *Proceedings of Institution of Chemical Engineering Research Event, Institution of Chemical Engineers,* 95- 97.

Storn, R. (1995). Differential Evolution design of an IIR- filter with requirements for magnitude and group delay. TR-95-018, *International Computer Science Institute.*

Upreti, S. R. & Deb, K. (1996). Optimal Design of an ammonia synthesis reactor using genetic algorithms. *Computers and Chemical Engineering, 21,* 87- 93.

Wang, F. S., Jing, C. H., and Tsao, G. T. (1998). Fuzzy- decision- making problems of fuel ethanol production using genetically engineered yeast. *Industrial & Engineering Chemistry Research, 37*(8), 3434- 3443.

Wolf, D. & Moros, R. (1997). Estimating rate constants of heterogeneous catalytic reactions without supposition of rate determining surface steps – An application of genetic algorithm. *Chemical Engineering Science, 52*, 1189-1199.