2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A Hybrid Surrogate Based Algorithm (HSBA) to Solve Computationally Expensive Optimization Problems

Hemant Kumar Singh, Amitay Isaacs and Tapabrata Ray

Abstract— **Engineering optimization problems often involve multiple objectives and constraints that are computed via computationally expensive numerical simulations. While the severe nonlinearity of the objective/constraint functions demand the use of population based searches (e.g. Evolutionary Algorithms), such algorithms are known to require numerous function evaluations prior to convergence and hence may not be viable in their native form. On the other hand, gradient based algorithms are fast and effective in identifying local optimum, but their performance is dependent on the starting point. In this paper, a hybrid algorithm is presented, which exploits the benefits offered by population based scheme, local search and also surrogate modeling to solve optimization problems with limited computational budget. The performance of the algorithm is reported on the benchmark problems designed for CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization.**

I. Introduction and Background

Evolutionary algorithms (EA) are population based metaheuristic methods that can be applied to solve unconstrained/constrained nonlinear optimization problems. Evolutionary Algorithms do not require gradient information and can be applied to solve non-linear, constrained, discontinuous/mixed integer as well as *black-box* functions. EAs search for global optimum solution(s) to optimization problems by searching different regions of the design space simultaneously.

In evolutionary algorithms, a population of candidate solutions is evolved over a number of generations to find the optimum solutions. Evolutionary algorithms are known to require evaluations of large number of solutions. Hence, for the design optimization problems requiring expensive simulations to evaluate the objective and the constraint functions, the total cost of the optimization can become quite prohibitive. Therefore, an important motivation exists to improve the efficiency and the effectiveness of evolutionary algorithms to reduce the computational cost of the optimization process.

To reduce the number of evaluations required for optimization, there are two key approaches in literature. The first approach relies on hybridization, i.e., a combination of a global search technique with a local search scheme. While local search algorithms are fast and effective, their performance is largely dependent on the starting point. Therefore, providing a good starting point for such algorithms is critical to find a good solution. In hybrid methods, a suitable global search method (such as EA) is used to identify good regions of search space, where the optimum is likely to be located. Thereafter, the local search is applied in order to refine the solution quickly. Such forms of algorithms can consist of a single cycle (global search followed by a local search), or multiple cycles (alternating global and local searches). This *hybrid* approach is also referred to as memetic algorithm [1]. For a review on memetic algorithms, the readers are referred to [2].

The second approach aims to reduce the computational cost of an optimization exercise through the use of approximations. Implied is that the objective and the constraint functions are approximated using certain functions. These replacement functions are referred to as surrogate models or metamodels. These surrogate models are computationally inexpensive to evaluate when compared to the simulations of the mathematical models and can be used in place of the expensive simulations. Even though surrogate models have being used within evolutionary algorithms, their use in not straightforward and requires consideration of many issues – type of the surrogate model, selection of the training data, global versus local models, and accuracy of the prediction – to name a few [3].

Function approximation involves building surrogate models that can approximate the response for the functions and are computationally cheaper than the original function evaluations. A generic function can be mathematically represented as $F(\mathbf{x}, y) = 0$, where $\mathbf{x}$ is a vector of m independent variables and y is the response. A surrogate model is trained using the responses $y_1, y_2, \ldots, y_N$ to solutions $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ sampled in the design space. The surrogate model is an explicit function of the form $y = f(\mathbf{x})$ that mimics the response y. The number of samples required to train the surrogate model is often related to the complexity of the function being approximated. The more complex the function, the more samples are required to adequately represent the response. Since the number of samples dictates the number of actual function evaluations, it is essential to keep the number of samples as low as possible for function evaluations requiring computationally expensive simulations. For a given number of samples, various sampling techniques try to position those samples in the design space to improve the quality of the surrogate model built using those samples.

There are many different types of surrogate models including Response Surface Methods (RSM) [4], Artificial Neural Networks (ANN) [5], Kriging [6], Support Vector Machines (SVM) [7] etc. In this work, a Kriging based surrogate model has been used. In the last decade, Kriging models have become quite popular due to their ability to

Hemant Kumar Singh and Tapabrata Ray are with the School of Engineering and Information Technology, University of New South Wales, Canberra, Australia (email: {h.singh,t.ray}@adfa.edu.au). Amitay Isaacs is with IBM Australia OzLabs, Canberra, Australia (email: amitay@gmail.com).

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

represent non-linear functions accurately. Kriging has been used with EAs to speed up the convergence for numerical test problems [8], [9], [10], [11] and in engineering applications including satellite boom optimization [12], piezoelectric actuator design [13], airfoil shape design [14], welded beam design [15].

In Kriging [16], the function of interest $y(\mathbf{x})$ is expressed as a combination of a global model and localized deviations:

$$y(\mathbf{x}) = f(\mathbf{x}) + Z(\mathbf{x}), \tag{1}$$

where $f(\mathbf{x})$ is a polynomial function and $Z(\mathbf{x})$ is a Gaussian model with mean 0 and variance σ^2. The co-variance matrix of $Z(\mathbf{x})$ is given by:

$$\text{Cov}[Z(\mathbf{x}^i), Z(\mathbf{x}^j)] = \sigma^2\mathbf{R} \tag{2}$$

where, $\mathbf{R}$ is a correlation matrix. Given a data set of n_s samples, the correlation between sample i and sample j is denoted by $\mathbf{R}(\mathbf{x}^i, \mathbf{x}^j)$. The matrix $\mathbf{R}$ is symmetric and of size $(n_s \times n_s)$. We assume a Gaussian correlation function with p_k set to 2.

$$\mathbf{R}(\mathbf{x}^i, \mathbf{x}^j) = \exp\left[-\textstyle\sum_{k=1}^{n_d} \theta_k |{x^i}_k - {x^j}_k|^{p_k}\right], \tag{3}$$

where n_d is the dimensionality of $\mathbf{x}$, θ_k and p_k are the hyper-parameters and ${x^i}_k$ and ${x^j}_k$ are the k^{th} components of $\mathbf{x}^i$ and $\mathbf{x}^j$ respectively. For a new point $\mathbf{x}^{n_s+1}$ where the prediction is sought, the approximated value $\hat{y}$ is given by the equation below:

$$\hat{y} = \hat{\mu} + \mathbf{r}^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}) \tag{4}$$

where, $\mathbf{y}$ is the column vector of size n_s containing the function values at the given data points and $\mathbf{1}$ is a column vector of all 1's.

$\hat{\mu}$ is estimated using the following equation:

$$\hat{\mu} = (\mathbf{1}^T\mathbf{R}^{-1}\mathbf{1})^{-1}\mathbf{1}^T\mathbf{R}^{-1}\mathbf{y} \tag{5}$$

$\mathbf{r}^T$ is a correlation vector of length n_s between the new point $\mathbf{x}^{n_s+1}$ and the n_s sampled data points:

$$\mathbf{r}^T = [\mathbf{R}(\mathbf{x}^{n_s+1}, \mathbf{x}^1), \ldots, \mathbf{R}(\mathbf{x}^{n_s+1}, \mathbf{x}^{n_s})]^T \tag{6}$$

The variance, $\hat{\sigma}^2$, can be estimated as follows:

$$\hat{\sigma}^2 = \tfrac{1}{n_s}\left[(\mathbf{y} - \mathbf{1}\hat{\mu})^T\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu})\right] \tag{7}$$

Finally, the correlation parameters, θ_k can be estimated by maximizing the likelihood and solving the following optimization problem:

$$\text{Maximize}\left(-\tfrac{1}{2}(n_s \ln(\hat{\sigma}^2) + \ln|\mathbf{R}|)\right) \tag{8}$$

The above optimization problem can be solved using a non-linear optimizer.

In this paper, a hybrid surrogate based evolutionary algorithm is presented to solve computationally expensive problems. The search proceeds in three successive phases: a) Global search, b) Kriging model building and search, and c) Local search. The algorithm is tested on the set of benchmark functions [17] proposed for *CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization* using a limited number of function evaluations.

Algorithm 1 Proposed Hybrid Surrogate Based Algorithm (HSBA)

Require: N {Population Size}
Require: $N_G > 1$ {Number of Generations}
Require: FE_{max} {Maximum Number of Function Evaluations}
$FE = 0$
$\mathbf{pop_1}$ = Initialize()
Evaluate_A($\mathbf{pop_1}$)
Update FE
$\mathbf{Archive}$ = $\mathbf{pop_1}$
for $i = 2$ to N_G **do**
  $\mathbf{childpop_{i-1}}$ = Evolve($\mathbf{pop_{i-1}}$)
  Evaluate_A($\mathbf{childpop_{i-1}}$) {True evaluation}
  Update $FE, \mathbf{Archive}$
  S=Rank($\mathbf{childpop_{i-1}}$ +$\mathbf{pop_{i-1}}$)
  $pop_i = S(1:N)$
end for
$\mathbf{xbest_{GA}}$ = Best solution obtained from GA
M=Generate-Model($\mathbf{Archive}$)
$\mathbf{xbest_{PS}}$ = Pattern search($\mathbf{xbest_{GA}}$)
Evaluate_A($\mathbf{xbest_{PS}}$) {True evaluation}
Update FE
$\mathbf{x_{0,LS}}$ = Better solution among $\mathbf{xbest_{GA}}$ and $\mathbf{xbest_{PS}}$
$\mathbf{xbest_{LS}}$ = Local search($\mathbf{x_{0,LS}}$) {Max. evals. allowed = $FE_{max} - FE$}
Update FE
if $FE < FE_{max}$ **then**
  $\mathbf{xbest}$ = Pattern search($\mathbf{xbest_{LS}}$) {Max. evals allowed = $FE_{max} - FE$}
else
  $\mathbf{xbest}$ = $\mathbf{xbest_{LS}}$
end if

II. Proposed Hybrid Surrogate Based Algorithm (HSBA)

The algorithm presented here attempts to capitalize on both surrogate model building as well as local search in order to solve computationally expensive problems with limited budget. The method is outlined in Algorithm 1, while the steps involved are discussed in the following subsections.

A. Global search

The first phase of the proposed algorithm is a global search using a population based stochastic algorithm (GA). A population size of $2 \times N_{var}$ is evolved for 10 generations on the true objective function. An archive of all solutions explored in the process is maintained. Thus, the archive contains all solutions evaluated using the true function. The best solution obtained from the search is referred to as $\mathbf{xbest_{GA}}$. The MATLAB inbuilt GA has been used in this phase with its default settings.

B. Kriging model generation and optimization

Once the global search is complete, a Kriging model for the problem is built with the solutions in the archive using the method described in Section I. For calculating the likelihood function, code available at [18] is used, while Matlab GA is used to maximize the likelihood function to construct the model. Thereafter, the best solution obtained from the global search ($\mathbf{xbest_{GA}}$) is used as a (potentially good) starting solution for an exhaustive Pattern search [19], [20] on the Kriging model (as opposed to true function evaluations). The best solution obtained from this search (in terms of predicted objective values from Kriging model), is then evaluated using the true function. This solution is referred to as ($\mathbf{xbest_{PS}}$).

C. Local search

Next, the better solution among $\mathbf{xbest_{GA}}$ and $\mathbf{xbest_{PS}}$ is referred to as $\mathbf{x_{0,LS}}$ and used as the starting solution for the local search. For performing the local search, *fmincon* function in Matlab is used. The final solution obtained after the local search is designated as $\mathbf{xbest_{LS}}$.

Lastly, *if* there are still leftover function evaluations, i.e., if the total number of function evaluations used in above three phases is less than the allowed number of function evaluations FE_{max}, a pattern search is used (starting from $\mathbf{xbest_{LS}}$) to further refine the solution obtained using the local search. The output of this step is the final $\mathbf{xbest}$ obtained by the algorithm.

III. Numerical Experiments

In this section, the performance of HSBA on the set of benchmark problems for CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization are presented. HSBA is implemented in Matlab 2009a.

A. Parameter settings

For each of the phases described in the algorithm in Section II, default functions available in Matlab 2009a are used. The functions are *ga* for the global search (GA), *patternsearch* for the Pattern search, and *fmincon* for the local search. Except for the population size ($2\times N_{var}$) and generations (10) for the global search phase, *none* of the algorithm parameters have been specified or adjusted. The algorithms are run with their default values in Matlab 2009a.

B. Results

The statistics of the results are calculated over 20 runs. The results reported are after FE_{max} evaluations, which is set to 500, 1000 and 1500 for 10, 20 and 30 variable problems respectively.

The convergence plots for a typical run are shown in Figures 1, 2 and 3 for 10, 20 and 30 variable problems respectively. In most plots, four distinct segments can be observed, which correspond to the improvement in the function value in each stage (global search, Kriging model optimization, local search using *fmincon* and pattern search). The last phase is only present for cases where total function evaluations for first three phases where less than FE_{max}.

C. Algorithm complexity

The relative time complexity ($\hat{T}_1/T_0$) of running the algorithm on different problems is summarized in Table II, where $\hat{T}_1$ is the average time taken for one run, and T_0 is the time taken to run the following routine in Matlab 2009a:

```
--------------------------------
for i=1:1000000
x= 0.55 + double(i);
x=x + x; x=x/2; x=x*x; x=sqrt(x);
x=log(x); x=exp(x); x=x/(x+2);
end
t0=toc(tstart);
--------------------------------
```

TABLE II: Average time complexity of HSBA for different problems across 20 runs

Problem	$\hat{T}_1/T_0$
1	864.306
2	7464.985
3	27972.132
4	784.309
5	7546.807
6	27511.990
7	828.780
8	7395.496
9	27487.388
10	651.977
11	7376.580
12	28238.899
13	543.936
14	6884.168
15	29643.927
16	690.278
17	7597.546
18	27732.008
19	812.133
20	7664.854
21	28439.405
22	703.126
23	7379.044
24	28139.415

IV. Summary and Conclusions

In this paper, a Hybrid Surrogate Based Algorithm (HSBA) is proposed to solve computationally expensive problems in very limited number of evaluations. The algorithm works by building a Kriging model obtained from a global search, and then further refines the solutions by doing a local search on the model as well as true problem. The performance of the algorithm is reported for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization. A comparison with other algorithms on the same problems is awaited from the special

TABLE I: Results obtained using HSBA after FE_{max} evaluations, for 20 runs

Problem	Best	Worst	Median	Mean	Std.
1	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
2	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
3	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
4	0.00000000	0.00000046	0.00000003	0.00000009	0.00000014
5	0.00000584	0.77935033	0.00481393	0.11144856	0.23943036
6	0.51373562	20.45091600	6.13638360	6.86658650	5.20101890
7	0.00000000	0.00000047	0.00000002	0.00000010	0.00000015
8	0.00327427	2.89277350	0.07208472	0.26583767	0.64039366
9	1.73164360	46.92244100	10.14122900	14.50943400	12.31920700
10	5.00000000	34.00000000	17.50000000	17.05000000	6.36168630
11	81.00000000	168.00000000	126.00000000	120.65000000	22.21965700
12	214.00000000	375.00000000	305.50000000	307.10000000	41.92713000
13	5.42414240	10.01616600	7.14840180	7.81596390	1.44906310
14	7.98453620	10.79675700	9.63006320	9.46549280	0.67411500
15	9.44224060	11.21212600	10.70945900	10.67134800	0.41202425
16	0.04921299	0.06398661	0.06398562	0.05807692	0.00742507
17	0.00000000	0.00000237	0.00000036	0.00000070	0.00000077
18	0.00000000	0.00000320	0.00000009	0.00000031	0.00000071
19	0.01255563	9.03805260	4.20116600	3.78768880	3.17235040
20	0.22826525	71.38461400	17.09644100	24.03333700	23.61827900
21	26.45026100	83.72679400	29.03959700	33.28979000	14.52148800
22	30.84367500	37.80831800	37.80831800	36.46513400	2.58797660
23	29.41721200	43.11223200	33.82310900	34.40343200	3.16913760
24	91.58095200	108.46712000	101.48667000	102.13593000	4.64321080

session, which will reveal the strengths and weaknesses of the proposed algorithm in dealing with different kind of single objective problems.

REFERENCES

[1] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," Caltech Concurrent Computation Program, Caltech, California, USA, Tech. Rep. C3P report 826, 1989.

[2] Y.-S. Ong, M. Lim, and X. Chen, "Memetic computation;past, present; future [research frontier]," *Computational Intelligence Magazine, IEEE*, vol. 5, no. 2, pp. 24 –31, may 2010.

[3] A. Isaacs, "Development of optimization methods to solve computationally expensive problems," Ph.D. dissertation, University of New South Wales, Australian Defence Force Academy, Canberra ACT, Australia, 2009.

[4] R. H. Myers and D. C. Montgomery, *Response Surface Methodology: Process and Product in Optimzation using Designed Experiments*. John Wiley & Sons, Inc., NY, USA, 1995.

[5] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1994.

[6] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, pp. 409–436, 1989.

[7] I. Steinwart and A. Christmann, *Support vector machines*. Springer, 2008.

[8] A. Ratle, "Accelerating the convergence of evolutionary algorithms by fitness landscape approximation," in *Parallel Problem Solving from Nature - PPSN V*, ser. Lecture Notes in Computer Science, A. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds., vol. 1498, 1998, pp. 87–96.

[9] ——, "Kriging as a surrogate finess landscape in evolutionary optimization," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 15, pp. 37–49, 2001.

[10] D. Büche, N. N. Schraudolph, and P. Koumoutsakos, "Accelerating evolutionary algorithms using fitness function models," in *Proceedings of GECCO Workshop on Learning, Adaptation and Approximation in Evolutionary Computation*, 2003, pp. 166–169.

[11] M. Li, G. Li, and S. Azarm, "A kriging metamodel assisted multi-objective genetic algorithm for design optimization," *Journal of Mechanical Design*, vol. 130, no. 3, p. 031401, Mar. 2008.

[12] M. A. El-Beltagy and A. J. Keane, "Evolutionary optimization for computationally expensive problems using gaussian processes," in *Proceedings of the International Conference on Artificial Intelligence*. CSREA, 2001, pp. 708–714.

[13] B. Wilson, D. Cappelleri, T. W. Simpson, and M. Frecker, "Efficient pareto frontier exploration using surrogate approximations," *Optimization and Engineering*, vol. 2, no. 1, pp. 31–50, Mar. 2001.

[14] M. Emmerich, A. P. Giotis, M. Özdemir, T. Bäck, and K. C. Giannakoglou, "Metamodel-assisted evolution strategies," in *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science. Springer, 2002, pp. 361–370.

[15] K. S. Won and T. Ray, "A framework for design optimization using surrogates," *Engineering Optimization*, vol. 37, no. 7, pp. 685–703, 2005.

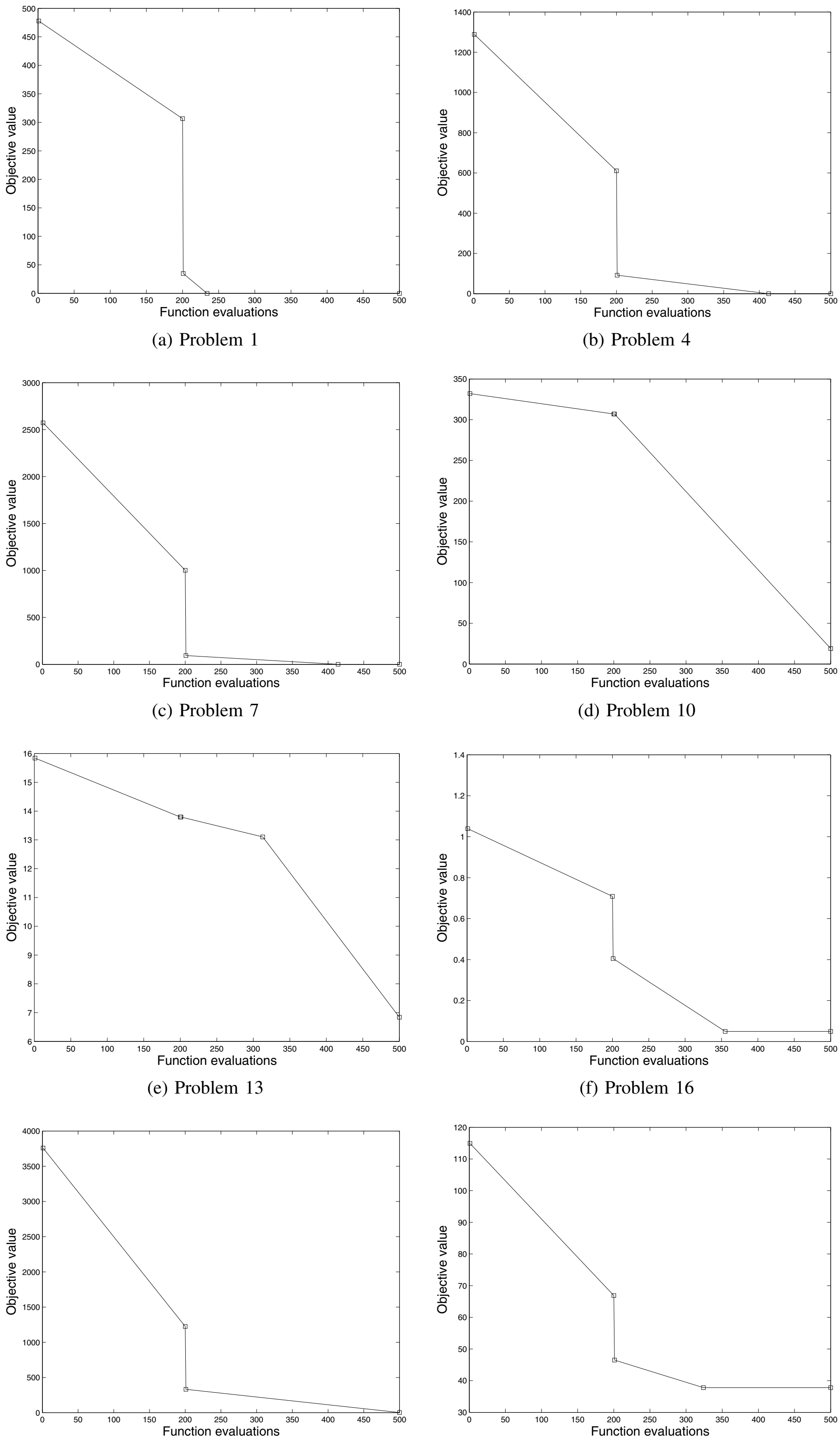
[16] T. W. Simpson, T. M. Mauery, J. J. Korte, and F. Mistree, "Kriging models for global approximation in simulation-based multidisciplinary design optimization," *AIAA journal, American Institute of Aeronautics and Astronautics*, vol. 39, no. 12, pp. 2233–2241, 2001.

[17] B. Liu, Q. Chen, Q. Zhang, J. J. Liang, P. N. Suganthan, and B. Y. Qu, "Problem definitions and evaluation criteria for computational expensive optimization," Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Nanyang Technological University, Singapore, Tech. Rep., 2013.

[18] A. Forrester. (2010) Engineering design via surrogate modelling: A practical guide (Matlab code). [Online]. Available: http://www.southampton.ac.uk/ aijf197/academic.htm

[19] T. G. Kolda, R. M. Lewis, and V. Torczon, "A generating set direct search augmented Lagrangian algorithm for optimization with a combination of general and linear constraints," Sandia National Laboratories, Tech. Rep., 2006.

[20] ——, "Optimization by direct search: New perspectives on some classical and modern methods," *SIAM review*, vol. 45, no. 3, pp. 385–482, 2003.



(a) Problem 1

(b) Problem 4

(c) Problem 7

(d) Problem 10

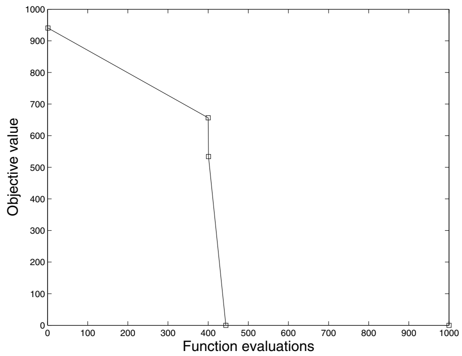
(e) Problem 13

(f) Problem 16

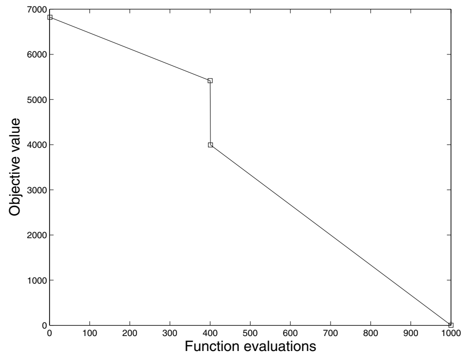
(g) Problem 19

(h) Problem 22

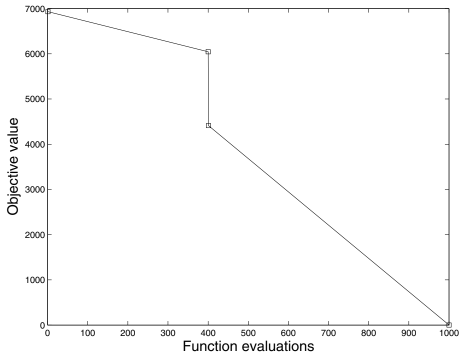
Fig. 1: Convergence plots for a typical run for 10 variable problems



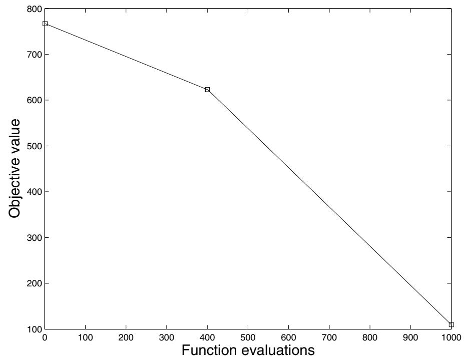
(a) Problem 2



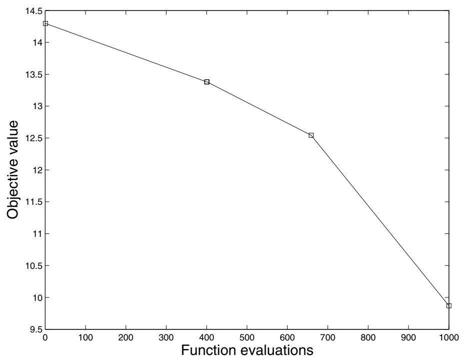
(b) Problem 5



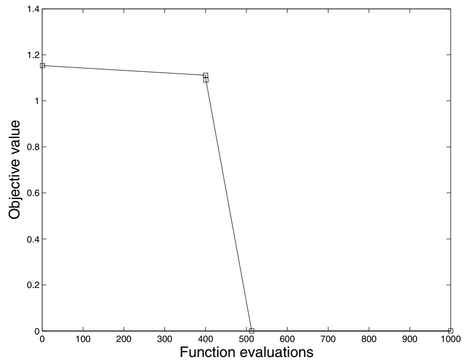
(c) Problem 8



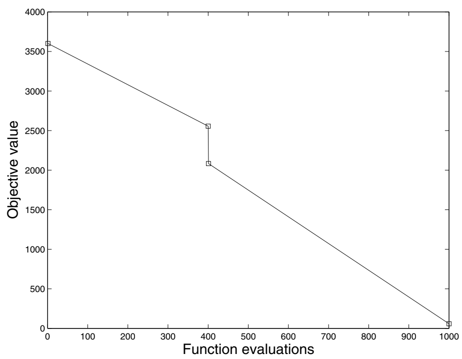
(d) Problem 11



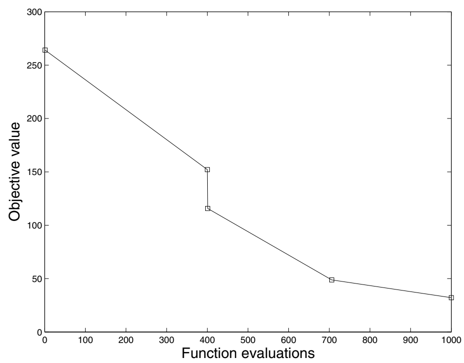
(e) Problem 14



(f) Problem 17

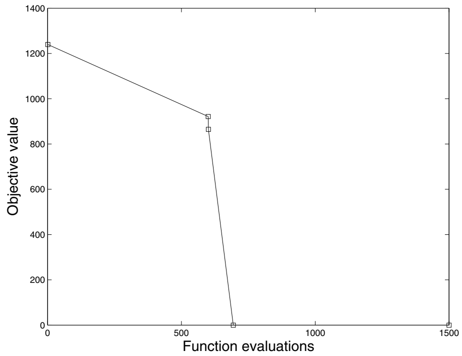


(g) Problem 20

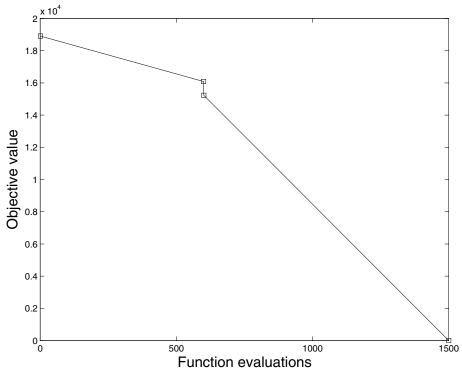


(h) Problem 23

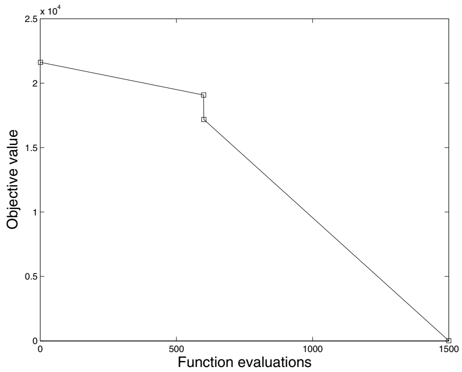
Fig. 2: Convergence plots for a typical run for 20 variable problems



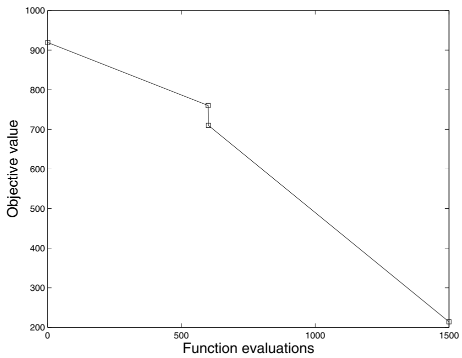
(a) Problem 3



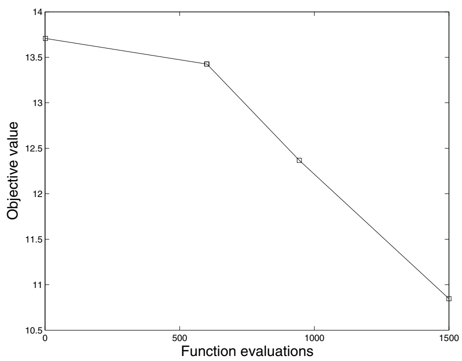
(b) Problem 6



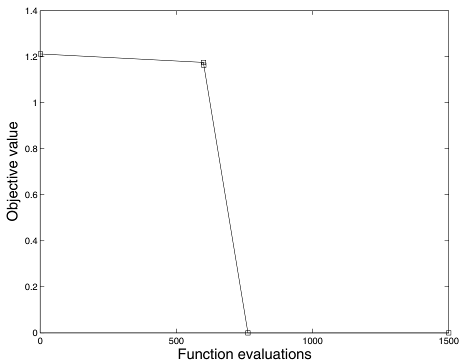
(c) Problem 9



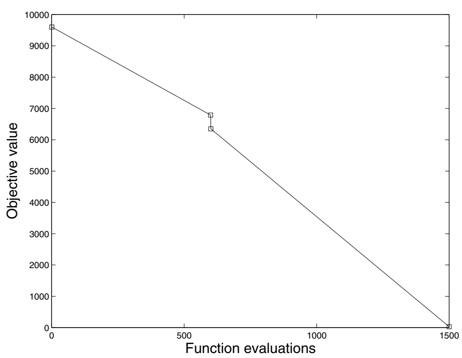
(d) Problem 12



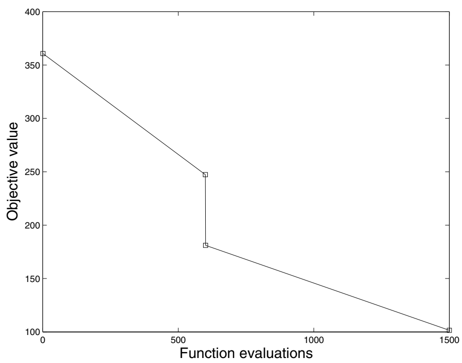
(e) Problem 15



(f) Problem 18



(g) Problem 21



(h) Problem 24

Fig. 3: Convergence plots for a typical run for 30 variable problems