Local Optimum Distance Evaluated Simulated Kalman Filter for Combinatorial Optimization Problems

Zulkifli Md Yusof, Ismail Ibrahim and Zuwairie Ibrahim
Faculty of Electrical and Electronics Engineering
Universiti Malaysia Pahang
26600 Pekan, Malaysia
zuwairie@ump.edu.my

Khairul Hamimah Abas
Faculty of Electrical Engineering
Universiti Teknologi Malaysia
81310 Skudai, Johor, Malaysia

Nor Azlina Ab Aziz, Nor Hidayati Abd Aziz
Faculty of Engineering and Technology
Multimedia University, 75450 Melaka
Malaysia

Mohd Saberi Mohamad
Faculty of Computing,
Universiti Teknologi Malaysia
81310 Johor, Malaysia

Abstract—Inspired by the estimation capability of Kalman filter, we have recently introduced a novel estimation-based optimization algorithm called simulated Kalman filter (SKF). Every agent in SKF is regarded as a Kalman filter. Based on the mechanism of Kalman filtering and measurement process, every agent estimates the global minimum/maximum. Measurement, which is required in Kalman filtering, is mathematically modelled and simulated. Agents communicate among them to update and improve the solution during the search process. However, the SKF is only capable to solve continuous numerical optimization problem. In order to solve combinatorial optimization problems, an extended version of SKF algorithm, which is termed as Local Optimum Distance Evaluated Simulated Kalman Filter (LODESKF), is proposed. Similar to existing approach, a mapping function is used to enable the SKF algorithm to operate in binary search space. A set of traveling salesman problems are used to evaluate the performance of the proposed LODESKF against DESKF

Keywords—simulated kalman filter; traveling salesman problem; combinatorial optimization

1. Introduction

In solving discrete optimization problems, algorithms such genetic algorithm (GA) [1], has been originally developed to operate in binary search space. However, not all optimization algorithms are originally developed to operate in binary search space. An example of these algorithms is simulated Kalman filter (SKF), which has been recently introduced by Ibrahim *et al.* in 2015 [2]. Since then, the SKF has been extended and employed to solve engineering problems [3-8]. In order to solve discrete optimization problems with SKF, modification or enhancement is needed. For example, sigmoid function has been employed as a mapping function to let gravitational search algorithm (GSA) to operate in binary search space [9]. The purpose of the mapping function is translated the velocity of GSA into probabilistic value. A random number is generated and compared with the probabilistic value in order to update the position of agent in binary search space.

There are a lot of discrete optimization problems in literature and real-world applications. Examples of discrete optimization problems are assembly sequence planning [10-11], DNA sequence design [12-13], VLSI routing [14-15], robotics drill route problem [16], and airport gate allocation problem [17]. Motivated by the importance of solving discrete optimization problems, the objective of this research is to modify the SKF algorithm for solving discrete optimization problems. However, unlike PSO, mapping function cannot be integrated in SKF because there is no specific variable in SKF that can be used as the input to mapping function. Nevertheless, the distance between an agent and the best agent can be exploited to let SKF operates in binary search space. This novel approach is introduced in this paper and this variant of SKF algorithm is called local optima distance evaluated SKF (DESKF). An interesting characteristic of this distance evaluated approach is that it is universal, which means that it can be integrated to any optimization algorithm such as PSO.

This paper is organized as follows. At first, SKF will be briefly reviewed followed by a detail description of the proposed distance evaluated SKF (DESKF) algorithm. Experimental set up will be explained and results will be shown and discussed. Lastly, a conclusion will be provided at the end of this paper.

2. SIMULATED KALMAN FILTER ALGORITHM

The simulated Kalman filter (SKF) algorithm is illustrated in Fig. 1. Consider n number of agents, SKF algorithm begins with initialization of n agents, in which the states of each agent are given randomly. The maximum number of iterations, t_{max}, is defined. The initial value of error covariance estimate, $P(0)$, the process noise value, Q, and the measurement noise value, R, which are required in Kalman filtering, are also defined during initialization stage.

Then, every agent is subjected to fitness evaluation to produce initial solutions $\{\boldsymbol{X}_1(0), \boldsymbol{X}_2(0), \boldsymbol{X}_3(0), \ldots, \boldsymbol{X}_{n-2}(0), \boldsymbol{X}_{n-1}(0), \boldsymbol{X}_n(0)\}$. The fitness values are compared and the agent having the best fitness value at every iteration, t, is registered as $\boldsymbol{X}_{\mathbf{best}}(t)$. For function minimization problem,

$$\boldsymbol{X}_{\mathbf{best}}(t) = \min_{i \in 1,\ldots,n} fit_i(\boldsymbol{X}(t)) \quad (1)$$

whereas, for function maximization problem,

$$\boldsymbol{X}_{\mathbf{best}}(t) = \max_{i \in 1,\ldots,n} fit_i\big(\boldsymbol{X}(t)\big) \quad (2)$$

The-best-so-far solution in SKF is named as $\boldsymbol{X}_{\mathbf{true}}$. The $\boldsymbol{X}_{\mathbf{true}}$ is updated only if the $\boldsymbol{X}_{\mathbf{best}}(t)$ is better (($\boldsymbol{X}_{\mathbf{best}}(t) < \boldsymbol{X}_{\mathbf{true}}$ for minimization problem, or $\boldsymbol{X}_{\mathbf{best}}(t) > \boldsymbol{X}_{\mathbf{true}}$ for maximization problem) than the $\boldsymbol{X}_{\mathbf{true}}$.

The subsequent calculations are largely similar to the predict-measure-estimate steps in Kalman filter. In the prediction step, the following time-update equations are computed.

$$\boldsymbol{X}_{\boldsymbol{i}}(t|t) = \boldsymbol{X}_{\boldsymbol{i}}(t) \quad (3)$$

$$P(t|t) = P(t) + Q \quad (4)$$

where $\boldsymbol{X}_i(t)$ and $\boldsymbol{X}_i(t|t)$ are the current state and current transition/predicted state, respectively, and $P(t)$ and $P(t|t)$ are the current error covariant estimate and current transition error covariant estimate, respectively. Note that the error covariant estimate is influenced by the process noise, Q.

The next step is measurement, which is a feedback to estimation process. Measurement is modelled such that its output may take any value from the predicted state estimate, $\boldsymbol{X}_i(t|t)$, to the true value, $\boldsymbol{X}_{\text{true}}$. Measurement, $\boldsymbol{Z}_i(t)$, of each individual agent is simulated based on the following equation:

$$\boldsymbol{Z}_{\boldsymbol{i}}(t) = \boldsymbol{X}_{\boldsymbol{i}}(t|t) + \sin(rand \times 2\pi) \times |\boldsymbol{X}_{\boldsymbol{i}}(t|t) - \boldsymbol{X}_{\boldsymbol{true}}| \quad (5)$$

The $\sin(rand \times 2\pi)$ term provides the stochastic aspect of SKF algorithm and $rand$ is a uniformly distributed random number in the range of $[0,1]$.

The final step is the estimation. During this step, Kalman gain, $K(t)$, is computed as follows:

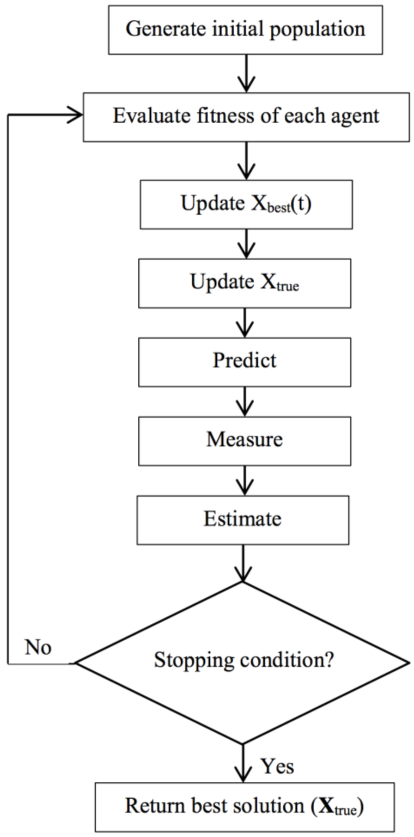


Figure-1. The simulated Kalman filter (SKF) algorithm.

$$K(t) = \frac{P(t|t)}{P(t|t)+R} \quad . \tag{6}$$

Then, the estimation of next state, $\boldsymbol{X}_i(t+1)$, is computed based on Eqn. (7).

$$\boldsymbol{X}_i(t+1) = \boldsymbol{X}_i(t|t) + K(t) \times (\boldsymbol{Z}_i(t) - \boldsymbol{X}_i(t|t)) \tag{7}$$

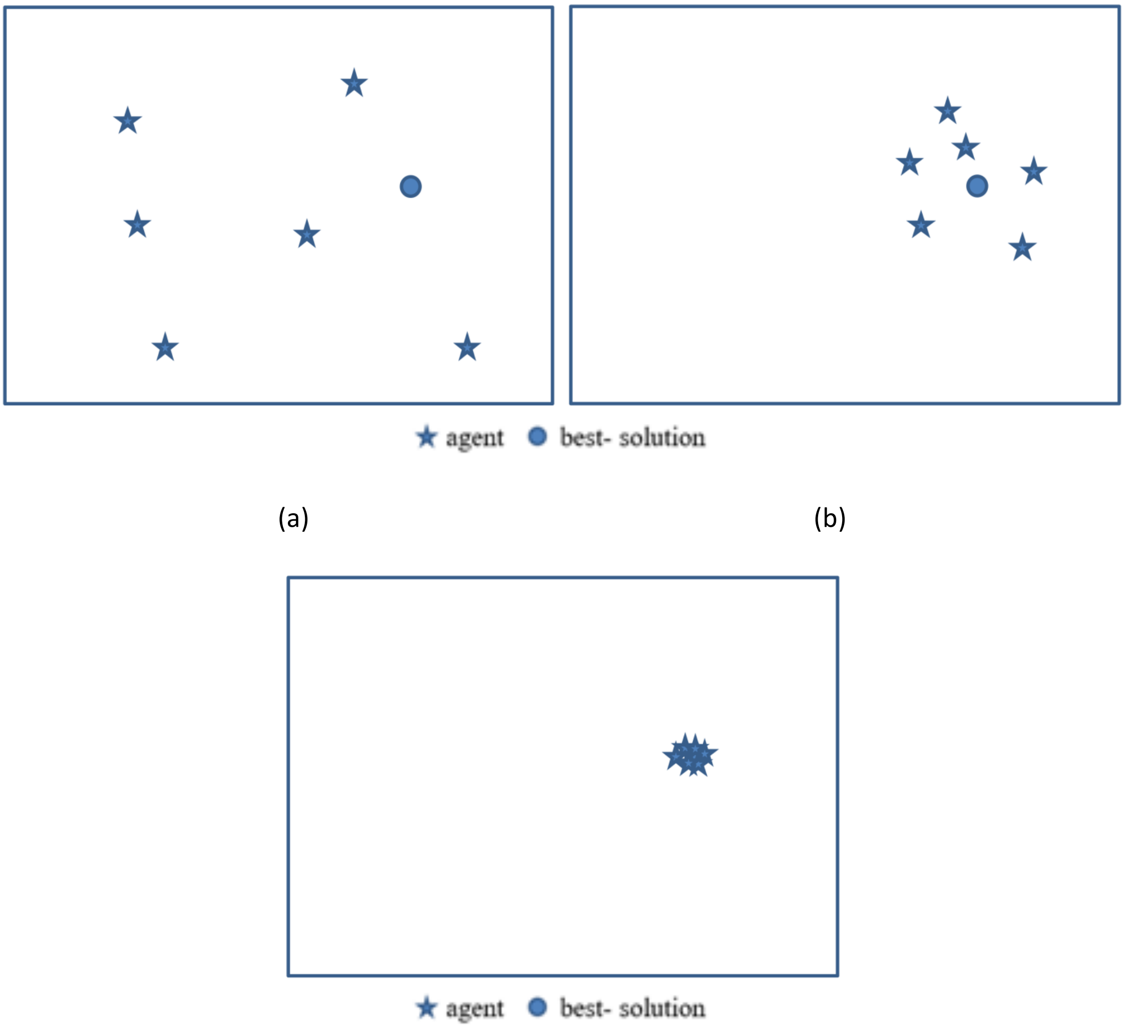
and the error covariant is updated based on Eqn. (8).

$$P(t+1) = \big(1 - K(t)\big) \times P(t|t) \tag{8}$$

Finally, the next iteration is executed until the maximum number of iterations, $t_{\max}$, is reached.

3. Local Optima Distance Evaluated Simulated Kalman Filter Alogrithm

In population-based search algorithm, generally, agents are randomly positioned in the search space. Then, the agents move in the search space to find global minimum or maximum. During the beginning of the search, exploration is preferred to make sure the search covers almost all regions in the search space. In this stage of search process, the position between agents is normally far with each other. As the search process continues, during the end of the search, exploration is no longer preferred because fine-tuning or exploitation is more preferred. During exploitation, agents becomes closer to each other and hence, the distance among them decreases.



(c)

Figure-2. Position of agents. (a) At the beginning of a search process. (b) During the middle of a search process. (c) At the end of a search process.

The position of agents in a search space during a typical search process is illustrated in Fig. 2. Normally, as the iteration continues, the distance between agents and the best-so-far solution decreases. This distance plays an important role in the proposed local optima distance evaluated simulated Kalman filter algorithm (LODESKF). In LODESKF, the distance is mapped into a probabilistic value [0,1] and then the probabilistic value will be compared with a random number [0,1] to update a bit string or solution to a combinatorial optimization problem.

In detail, most of the calculations in the proposed LODESKF are similar to the original SKF. Modifications are needed only during initialization and generation of solution to combinatorial optimization problem.

A. Initialization

During the initialization of agents, in SKF, the states of each agent are given randomly. An additional initialization is introduced in LODESKF. Every agent is associated with a random bit string as well. The length of the bit string is problem dependent and subjected to the size of the problem. Thus, 2 types of variables are associated with an agent in SKF. They are continuous variable, $\boldsymbol{x}$, which is produced as estimated value of SKF (also similar to the position of agents in a search space), and a bit string, $\boldsymbol{\Sigma}$, which is used to represent solution to a combinatorial optimization problem.

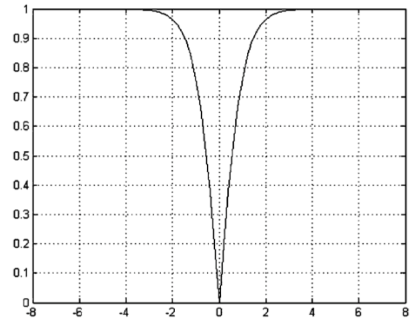


Figure-3. A probabilistic function used in [9]. Note that y-axis is the probabilistic value and x-axis is the distance.

B. Generation of Solution to a combinatorial optimization problem

In LODESKF, for a particular *d*th dimension, the distance between an *i*th agent to the best solution at iteration *t* can be calculated as follows:

$$D_i^d(t) = x_i^d(t) - x_{best}^d(t) \tag{9}$$

In binary gravitational search algorithm (BGSA) [9], a function, as shown in Fig. 5, is used to map a velocity value into a probabilistic value within interval [0,1]. Similar function is used in LODESKF. This distance value, $D_i^d(t)$, is mapped to a probabilistic value within interval [0,1] using a probability function, $S(D_i^d(t))$, as follows:

$$S\left(D_i^d(t)\right) = \left|\tanh\left(D_i^d(t)\right)\right| \tag{10}$$

After the $S\left(D_i^d(t)\right)$ is calculated, a random number, *rand*, is generated and a binary value at dimension *d* of an *i*th agent, Σ_i^d, is updated according to the following rule:

if $\quad rand < S\left(D_i^d(t)\right)$

then $\Sigma_i^d(t+1) = \text{complement}\Sigma_i^d(t+1)$

else $\Sigma_i^d(t+1) = \ \Sigma_i^d(t+1)$

end

4. EXPERIMENTS, RESULT AND DISCUSSION

The LODESKF is applied to solve a set of TSP. The objective of TSP is to find the shortest distance from a start city to an end city while visiting every city not more than once. In this paper, 28 instances of TSPs are considered, from the size of 51 cities to 2103 cities, as shown in Table 1. These problems were taken from TSPLib [18].

Experimental setting for LODESKF is shown in Table 2. For benchmarking purpose, additional experiments were considered, which are based on the Distance Evaluated Simulated Kalman Filter (DESKF) [3]. Experimental setting for LODESKF is identical to DESKF as in Table 2. In all experiments, the number of runs, the number of agents, and the number of iterations are 50, 30, and 1000, respectively.

The proposed LODESKF is compared with DESKF. The average performances of the two algorithms are presented in Table 3. The numbers written in bold show the best performance. The standard deviations are tabulated at Table 4.

Based on these average performances, Wilcoxon signed rank test is performed. The result of the test is tabulated in Table 5. The level of significant chosen here is $\sigma = 0.05$. It is found that statistically no significant difference is found between the proposed LODESKF and DESKF. Both of the algorithms perform as good as each other in solving TSP problems. However, statistically, DESKF is found to perform significantly better than LODESKF in solving the benchmark problems used in this work. Examples of convergence curves are shown in Fig. 4 to Fig. 6.

Table-1. Property of the test problems.

TSP Index	Name	Size	TSP Index	Name	Size
1	Berlin52	52	17	KROE100	100
2	Bier127	127	18	LIN105	105
3	Ch130	130	19	LIN318	318
4	Ch150	150	20	P654	654
5	D198	198	21	PCB442	442
6	D493	493	22	PR76	76
7	D657	657	23	PR107	107
8	D1291	1291	24	PR124	124
9	DSJ1000	1000	25	PR136	136
10	Eil51	51	26	PR144	144
11	Eil76	76	27	PR152	152
12	Eil101	101	28	PR226	226
13	FL1400	1400	29	PR264	264
14	KROB200	200	30	PR299	299
15	KROC100	100	31	PR439	439
16	KROD100	100			

Table-2. Experimental setting parameters.

SKF parameters	
Parameter	**Value**
P	1000
Q	0.5
R	0.5
rand	[0,1]
$x_{\min}$	-100
$x_{\max}$	100

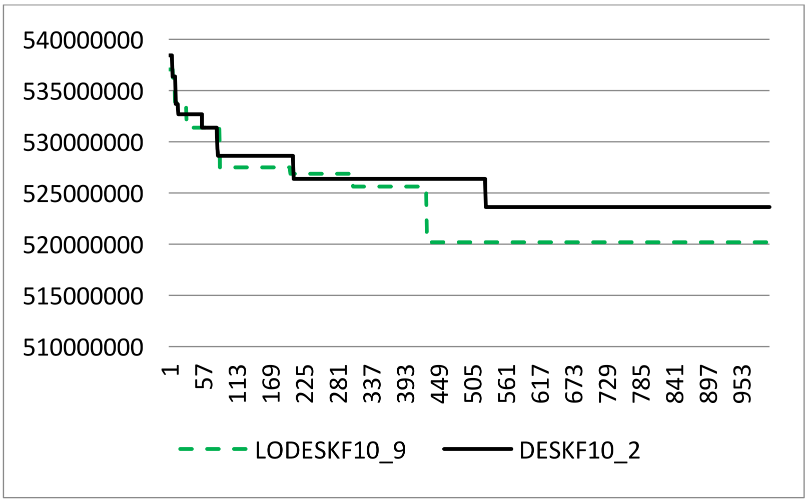


Figure-4. An example of convergence curve for TSP index 10 (DSJ1000).
Note that y-axis is the fitness value and x-axis is the iteration value.

Table 3: Average performance

CASE	LODESKF	DESKF
Berlin52	23045.1319	**22932.1956**
Bier127	545900.741	**544106.719**
Ch130	39182.5483	**39254.3698**
Ch150	**46158.5**	46270.792
D198	157762.888	**157618.449**
D493	412276.375	**411998.897**
D657	**794297.409**	796175.26
D1291	**1643205.37**	1645013.36
DSJ1000	524029496.9	**524027900.0**
Eil51	1276.72676	**1268.41674**
Eil76	**2046.52214**	2052.85533
Eil101	2847.57036	**2845.65913**
FL1400	**1580657.97**	1581880.82
KROB200	286155.227	**285802.695**
KROC100	136327.93	**135469.486**
KROD100	132213.627	**131622.265**
KROE100	**137976.013**	138503.846
LIN105	**98523.6012**	99036.1944
LIN318	**526154.129**	527049.445
P654	1846235.79	**1845491.98**
PCB442	**707113.303**	708486.345
PR76	883878.917	**461023.282**
PR107	884176.464	**446386.828**
PR124	885729.758	**580257.803**
PR136	884774.385	**690108.269**
PR144	885865.667	**682605.349**
PR152	**886087.79**	886217.152
PR226	**1478185.98**	1479082.48
PR264	**952318.762**	954199.001
PR299	665725.863	**664536.983**
PR439	**1733255.36**	1737005.43

Table 4: Standard Deviation performance

CASE	LODESKF	DESKF
Berlin52	499.5224	603.6889
Bier127	6611.013	7281.643
Ch130	489.4423	516.4827
Ch150	589.9976	569.2936
D198	2616.943	2897.983
D493	3031.436	3280.208
D657	4808.49	4105.962
D1291	5819.507	5144.381
DSJ1000	2478966	2337073.3
Eil51	28.57204	31.57152
Eil76	35.10814	34.06516
Eil101	50.25994	38.1741
FL1400	8198.207	7746.364
KROB200	3439.746	3467.673
KROC100	2697.375	2856.467
KROD100	2245.638	1932.355
KROE100	3055.123	2382.471
LIN105	2197.825	1895.213
LIN318	4532.086	4245.346
P654	13698.44	11723.48
PCB442	5509.457	5458.947
PR76	15020.29	9421.58
PR107	12779.57	9305.757
PR124	11745.28	9098.877
PR136	12168.75	10382.35
PR144	10924.8	8146.23
PR152	11559.43	11991.31
PR226	15755.83	17767.7
PR264	13527.21	10336.98
PR299	8275.882	5870.534
PR439	16539.96	14461.78

5. Conclusion

This paper reports the another attempt to use SKF for solving combinatorial optimization problems. Based on the proposed LODESKF, the distance between an agent to the best solution is evaluated to update a binary value. Experimental result and analysis showed the potential of LODESKF. Even though the performance of DESKF is better than LODESKF. Currently, more experiments are being done. Also, various TSP instances are considered in order to observe a more concrete conclusion.

Table 5: Ranking Using WilcoxonTest.

CASE	DESKF	LODESKF	d value	d sign	rank -ve	rank +ve
Eil101	2845.659133	2847.570361	1.911228	1	0	1
Eil76	2052.855332	2046.52214	-6.33319	-1	2	0
Eil51	1268.416735	1276.726756	8.310021	1	0	3
Ch130	39254.36981	39182.54831	-71.8215	-1	4	0
Ch150	46270.79203	46158.49999	-112.292	-1	5	0
Berlin52	22932.19562	23045.13194	112.9363	1	0	6
PR152	886217.1518	886087.7898	-129.362	-1	7	0
D198	157618.4494	157762.888	144.4386	1	0	8
D493	411998.8972	412276.375	277.4778	1	0	9
KROB200	285802.6951	286155.2267	352.5316	1	0	10
LIN105	99036.19444	98523.60122	-512.593	-1	11	0
KROE100	138503.8464	137976.0125	-527.834	-1	12	0
KROD100	131622.2648	132213.6272	591.3624	1	0	13
P654	1845491.981	1846235.788	743.8071	1	0	14
KROC100	135469.4863	136327.93	858.4437	1	0	15
LIN318	527049.4451	526154.1292	-895.316	-1	16	0
PR226	1479082.478	1478185.981	-896.497	-1	17	0
PR299	664536.9829	665725.8625	1188.88	1	0	18
FL1400	1581880.821	1580657.972	-1222.85	-1	19	0
PCB442	708486.3455	707113.3028	-1373.04	-1	20	0
DSJ1000	524027900.0	524029496.9	1596.93	1	0	21
Bier127	544106.7191	545900.741	1794.022	1	0	22
D1291	1645013.358	1643205.369	-1807.99	-1	23	0
D657	796175.2598	794297.4094	-1877.85	-1	24	0
PR264	954199.0008	952318.7618	-1880.24	-1	25	0
PR439	1737005.429	1733255.364	-3750.07	-1	26	0
PR136	690108.2687	884774.3855	194666.1	1	0	27
PR144	682605.3485	885865.6675	203260.3	1	0	28
PR124	580257.8035	885729.7577	305472	1	0	29
PR76	461023.2818	883878.9165	422855.6	1	0	30
PR107	446386.8279	884176.4638	437789.6	1	0	31
					211	285

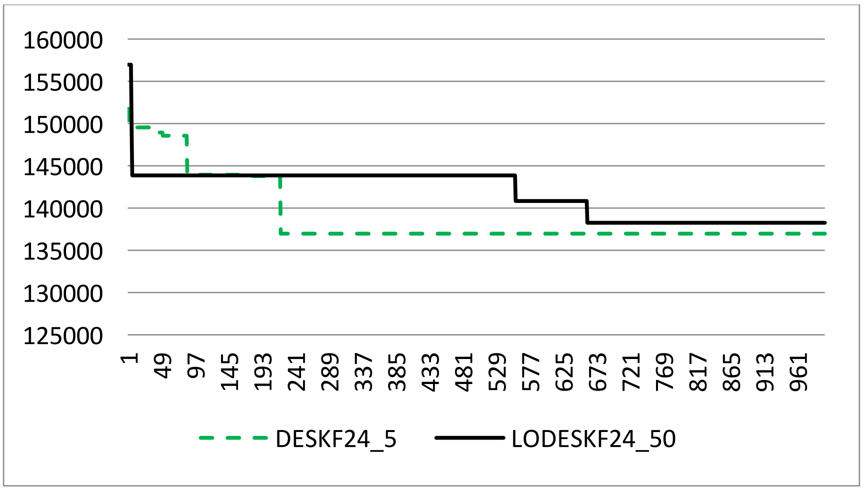


Figure-5. An example of convergence curve for TSP index 24 (KROE100). Note that y-axis is the fitness value and x-axis is the iteration value

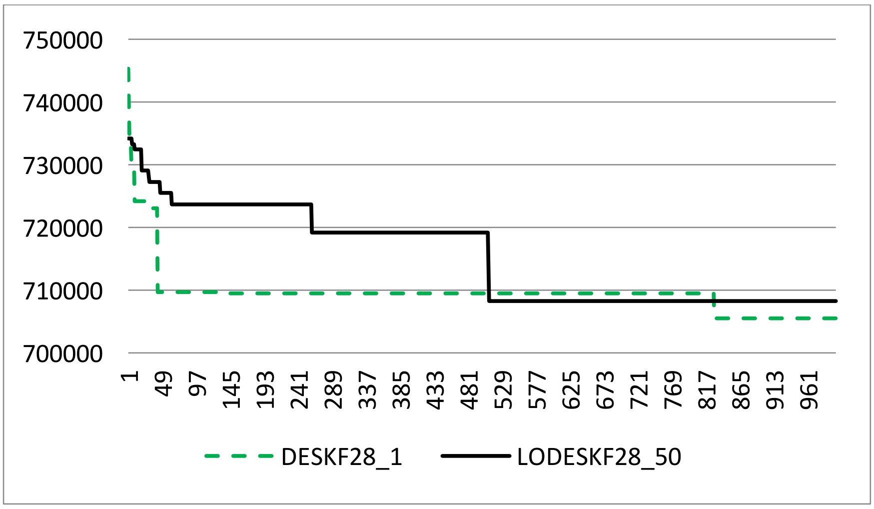


Figure-6. An example of convergence curve for TSP index 28 (PCB442). Note that y-axis is the fitness value and x-axis is the iteration value

ACKNOWLEDGMENT

The authors will like to thank Universiti Malaysia Pahang for providing internal financial support through grant GRS1503120. This research is also supported by Fundamental Research Grant Scheme (FRGS) awarded to Universiti Malaysia Pahang (RDU).

REFERENCES

[1] D. E. Goldberg. Genetic algorithm in search, optimization and machine learning. Addison-Wesley Longman Publishing Co., Inc. Boston, USA. 1989

[2] Z. Ibrahim., N. H. A. Aziz, N. A. A. Aziz, S. Razali, M. I. Shapiai, S. W. Nawawi, and M. S. Mohamad, “A Kalman filter approach for solving unimodal optimization problems,” ICIC Express Lett., vol. 4, no. 5, pp. 1–8, 2015.

[3] Z. Md Yusof, Z. Ibrahim, I. Ibrahim, K. Z. Mohd Azmi, N. A. Abd Aziz, N. H. Abd Aziz, and M. S. Mohamad, Distance Evaluated Simulated Kalman Filter for Combinatorial Optimization Problems, ARPN Journal of Engineering and Applied Sciences, Vol. 11, No. 7, pp. 4904-4910, 2016.

[4] Z. Md Yusof, Z. Ibrahim, I. Ibrahim, K. Z. Mohd Azmi, N. A. Ab Aziz, N. H. Abd Aziz, and M. S. Mohamad, “Angle modulated simulated kalman filter algorithm for combinatorial optimization problems,” ARPN J. Eng. Appl. Sci., vol. 11, no. 7, pp. 4854–4859, 2016.

[5] B. Muhammad, Z. Ibrahim, K. H. Ghazali, K. Z. Mohd Azmi, N. A. Ab Aziz, N. H. Abd Aziz, and M. S. Mohamad, “A new hybrid simulated Kalman filter and particle swarm optimization for continuous numerical optimization problems,” ARPN J. Eng. Appl. Sci., vol. 10, no. 22, pp. 17171–17176, 2015.

[6] Z. Md Yusof, I. Ibrah, and S. N. Satiman, “BSKF : Binary Simulated Kalman Filter,” in Third International Conference on Artificial Intelligence, Modelling and Simulation, pp. 77–81, 2015

[7] Z. Md Yusof, S. N. Satiman, K. Z. Mohd Azmi, B. Muhammad, S. Razali, Z. Ibrahim, Z. Aspar, and S. Ismail, “Solving Airport Gate Allocation Problem using Simulated Kalman Filter,” Int. Conf. Knowl. Transf., 2015.

[8] A. Adam, Z. Ibrahim, N. Mokhtar, M. I. Shapiai, M. Mubin, I. Saad, “Feature selection using angle modulated simulated Kalman filter for peak classification of EEG signals, SpringerPlus,” Vol. 5, No. 1580, 2016.

[9] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi. BGSA: binary gravitational search algorithm. Natural Computing, Vol. 9, Issue 3, pp. 727-745, 2010

[10] J.A.A. Mukred, Z. Ibrahim, I. Ibrahim, A. Adam, K. Wan, Z. Md Yusof, and N. Mokhtar. A Binary Particle Swarm Optimization Approach to Optimize Assembly Sequence Planning, Advance Science Letters, Vol. 13, pp. 732-738, 2012

[11] I. Ibrahim, Z. Ibrahim, H. Ahmad, M.F. Mat Jusof, Z. Md Yusof., S.W. Nawawi, and M. Mubin. An Assembly Sequence Planning Approach with a Rule-based Multi State Gravitational Search Algorithm, The International Journal of Advanced Manufacturing Technology, Vol. 79, pp. 1363-1376, 2015

[12] F. Yakop, Z. Ibrahim, A.F. Zainal Abidin, Z. Md Yusof., M.S.Mohamad, K. Wan K., and J. Watada. An Ant Colony System for Solving DNA Sequence Design Problem in DNA Computing, International of Innovative Computing, Information and Control, Vol. 8, No. 10, pp. 7329-7339, 2012.

[13] Z. Ibrahim, N.K. Khalid, I. Ibrahim, K. S. Lim, S. Bunyamin, Z. Md Yusof., and M.S. Muhammad. Function Minimization in DNA Sequence Design Based on Binary Particle Swarm Optimization. Journal Teknologi D (UTM), No. 54, pp. 331-342, 2011.

[14] Z. Md Yusof.,., A.F. Zainal Abidin, M.N.A. Salam, K. Khalil, J.A.A. Mukred, M.K. Hani, and Z. Ibrahim. A Binary Particle Swarm Optimization Approach for Buffer Insertion in VLSI Routing. International Journal of Innovative Management, Information and Production, Vol. 2, No. 3, pp. 34-39. 2011.

[15] Z. Md Yusof., A.F. Zainal Abidin, A. Adam, K. Khalil, J.A.A. Mukred, M.S. Mohamad, M.K. Hani, and Z. Ibrahim. A Two Step Binary Binary Particle Swarm Optimization Approach for Routing in VLSI. ICIC Express Letters, Vol. 6, No. 3, pp.771-776, 2012.

[16] M.H. Othman, A.F. Zainal Abidin, A. Adam, Z. Md Yusof., Z. Ibrahim, S.M. Mustaza, and Y.Y. Lai. A Binary Particle Swarm Optimization Approach for Routing in PCB Holes Drilling Process, International Conference on Robotic Automation System (ICORAS2011), pp. 201-206, 2011.

[17] A. Bourals, M.A. Ghaleb, U.S. Suryahatmaja, and A.M. Salem 2014. The Airport Gate Assignment Problem: A Survey. The Scientific World Journal, 2014

[18] comopt.ifi.uni-heidelberg.de/software/TSPLIB95/