Continuous Population-Based Incremental Learning with Mixture Probability Modeling for Dynamic Optimization Problems

Adrian Lancucki[1], Jan Chorowski[1], Krzysztof Michalak[2], Patryk Filipiak[1], and Piotr Lipinski[1]

[1] Computational Intelligence Research Group, Institute of Computer Science, University of Wroclaw, Wroclaw, Poland,
{adrian.lancucki, jan.chorowski, patryk.filipiak, piotr.lipinski}@cs.uni.wroc.pl
[2] Institute of Business Informatics, Wroclaw University of Economics, Wroclaw, Poland
krzysztof.michalak@ue.wroc.pl

Abstract. This paper proposes a multimodal extension of $\mathrm{PBIL_C}$ based on Gaussian mixture models for solving dynamic optimization problems. By tracking multiple optima, the algorithm is able to follow the changes in objective functions more efficiently than in the unimodal case. The approach was validated on a set of synthetic benchmarks including Moving Peaks, dynamization of the Rosenbrock function and compositions of functions from the IEEE CEC'2009 competition. The result obtained in the experiments proved the efficiency of the approach in solving dynamic problems with a number of competing peaks.

Keywords: evolutionary algorithms, estimation of distribution algorithms, dynamic optimization problems, multimodal optimization

1 Introduction

Many real-world optimization problems have a dynamic character, often defined as the tendency of the objective function or the constraints to change as time goes by. However, when virtually any parameter can be variable over time, reliable evaluation of algorithms becomes even harder than with static problems. It might be the case that, when solving real-world problems, one might benefit from multimodality. Tracking multiple optima and actively searching for emerging ones would enable reacting instantly to disruptive changes of objective function.

Estimation of distribution algorithms (EDAs) employ probabilistic models instead of traditional genetic operators. Typically, at each generation a new population is drawn from the distribution represented by the model, which is afterwards updated based on the fittest specimen. Thus, EDAs construct probability distributions describing good solutions, while carrying the search for optima. EDAs are characterized by the family of those distributions. The model can be calculated based on frequency of particular genes [4]. Often, estimation of one-dimensional distributions takes place, as in UMDA [9, 13]. In [14] estimation of two variables has been proposed. More advanced models include: decision trees [18], one- and multidimensional Gaussian models [17, 5], Boltzmann Machines [12], Bayesian Networks [8].

Applications of EDAs to DOPs have been little explored [13, 17]. Yang et al. [15] proposed maintaining diversity in the population by using two probabilistic models; the former optimizes solutions, while the latter is a model with high variance; mechanisms similar to random immigrants were also investigated. In [17] an approach is mentioned that controls the pace of convergence of the population through adjusting the Gaussian model proposed in [16], where the additional model is being activated, when the main model is not able to generate good solutions.

Despite the benefits, EDAs are not yet effective at solving DOPs. Probabilistic modeling techniques are either too simple, ignoring intrinsic dependencies, or too complex for successful estimation of density based on a fairly small population. Conversely, EDAs which take into account dependencies between random variables are usually too computationally demanding to keep up with changes in the objective function. Most of EDAs model unimodal probability distributions, focusing on a particular area of the search space. However, broad monitoring of optimization landscape changes is essential to dynamic optimization.

Recently, there were studies on applications of mixture models, which enable modeling of multimodal distributions, and therefore facilitate solving optimization problems with many similar local optima. This paper proposes a novel approach to solving DOPs with multimodal EDAs.

2 Algorithm

Multimodality in an EDA might be promoted by modeling the problem at hand with a mixture model. This publication proposes to employ the Gaussian mixture model; with M modes, pdf of distribution given by the mixture is

$$p(\boldsymbol{x}) = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma}_i|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^\intercal \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i)\right), \qquad (1)$$

where $\boldsymbol{\Sigma}$ is diagonal (elements σ_{jj} are referred to simply as σ_j). Such distribution is used at every generation to draw the entire population, the model is then adapted to the population based on its fitness.

This paper proposes a new algorithm based on PBIL$_\mathrm{C}$ [11], Multimodal Continuous Population-Based Incremental Learning (MPBIL$_\mathrm{C}$), for approaching DOPs. In the original work PBIL$_\mathrm{C}$ has been applied to continuous domains by modeling gene distributions with normal distribution. Proposed extension incorporates mixtures of Gaussians. M models of $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ are maintained at all times, allowing to model sub-populations focused on different local optima, which might become global in subsequent timeslices. Update rules parametrized with $(\alpha, \beta, \delta, \rho)$, similar to those in [11], govern updates of $\boldsymbol{\Sigma}_i$ and $\boldsymbol{\mu}_i$:

$$\boldsymbol{\mu}_i^{(t+1)} = (1-\alpha)\boldsymbol{\mu}_i^{(t)} + \alpha(\boldsymbol{P}_i^1 - \boldsymbol{P}_i^N),$$
$$\boldsymbol{\Sigma}_i^{(t+1)} = \max\left(\delta,\ (1-\beta)\boldsymbol{\Sigma}_i^{(t)} + \beta\cdot\mathrm{stdev}(\underbrace{\boldsymbol{P}_i^1,\dots,\boldsymbol{P}_i^{\lceil\rho\cdot\mathrm{count}(i)\rceil}}_{\lceil\rho\cdot\mathrm{count}(i)\rceil})\right), \tag{2}$$

where $\boldsymbol{P}_i^j$ is the jth fittest individual among the ones assigned to the ith Gaussian, and stdev$(\cdot)$ returns a diagonal matrix of standard deviations.

Pseudocode for MPBIL$_\mathrm{C}$ is shown in Algorithm 1. Gaussians are initialized randomly in the search space with $\boldsymbol{\Sigma}_{init}$; with each timeslice, their $\boldsymbol{\mu}_i$ parameters are maintained while setting $\boldsymbol{\Sigma}_i$ back to $\boldsymbol{\Sigma}_{init}$. During sub-evolution, consecutive populations are drawn at each generation with the underlying Gaussian mixture. After evaluation, each individual is being assigned to the closest Gaussian, and their parameters are being updated. Gaussians are expected to iteratively converge towards Dirac delta function and halt with all σs equal to δ.

Algorithm 1 Pseudocode of the MPBIL$_\mathrm{C}$ algorithm

$M \leftarrow InitializeModels(N, \boldsymbol{\Sigma}_{init})$
while not $TerminationCondition$ **do** ▷ For each timeslice
  for $t \leftarrow 1$ **to** N_{gen} **do** ▷ Begin sub-evolution
    $P \leftarrow DrawPopulation(M)$
    $EvaluatePopulation(P)$
    if $t \equiv 0 \pmod{t_r}$ **then**
      $EvaluateModels(M)$
      $m_w \leftarrow WorstModel(M)$
      $RandomlyInitializeModel(M, m_w, \boldsymbol{\Sigma}_r)$
    end if
    $AssignIndividualsToModels$
    $M \leftarrow AdaptModels(M, \alpha, \beta, \delta, \rho)$
  end for
  $ResetModelsStdev(\boldsymbol{\Sigma}_{init})$
end while

Depending on the objective, it is possible for many modes to converge to the same optima. A mechanism of randomly scattering the Gaussians, governed by parameters $(t_r, \boldsymbol{\Sigma}_r)$ prevents it. We measure the importance of each mixture's component by computing its overlap with other components:

$$\sum_{j=1}^{M} \exp\left(-\prod_{l=1}^{d}\left(\frac{\boldsymbol{\mu}_i^{(t)}[l] - \boldsymbol{\mu}_j^{(t)}[l]}{\boldsymbol{\sigma}_i^{(t)}[l]}\right)^2\right), \tag{3}$$

where $\boldsymbol{\mu}[l]$ denotes lth element of $\boldsymbol{\mu}$. Every t_r iterations the Gaussian with the highest overlap with other mixture components is reinitialized in a random search space location with $\boldsymbol{\Sigma}_r$. Those parameters have to be chosen with care: t_r should be large enough to allow modes to converge, while $\boldsymbol{\Sigma}_r$ should reflect the size of the search space. Large values create an adverse effect of scattering individuals across all other Gaussians.

3 Validation of the Approach

3.1 Performance Measures

To assess performance of the proposed algorithm, different measures were employed during experiments. Nguyen et al. [7] summarizes various measures for DOPs; not all of them are feasible for EDAs, and some require full knowledge of the objective function (location of optima). The most general ones were chosen, with emphasis on measuring multimodality:

$$
\begin{array}{llll}
m_1: & E_{MO} = \frac{1}{n}\sum_{j=1}^{n} e_{MO}(j) & m_6: & PC_{err}^{(t)} = \frac{1}{G}\sum_{i=1}^{G} \frac{\sum_{j=1}^{\#opt} err_{best(j)}^{(t)}}{\#opt} \\
m_2: & E_B = \frac{1}{m}\sum_{i=1}^{m} e_B(i) & m_7: & err_{best}^{(t)} = \frac{1}{N}\sum_{i=1}^{N} F(best_{EA}^{(t)}) - Min_F^{(t)} \\
m_3: & I = \sum_{i=1}^{N}\sum_{j=1}^{P}(x_{ij} - c_i)^2 & m_8: & \bar{F}_{BOG} = \frac{1}{G}\sum_{i=1}^{G}\left(\frac{1}{N}\sum_{j=1}^{N} F_{BOG_{ij}}\right) \\
m_4: & explr^{(t)} = \frac{\sum_{i=1}^{N} dist(c_i, i)\times f(i)}{N} & m_9: & stab_{F,EA}^{(t)} = \max\{0, acc_{F,EA}^{(t-1)} - acc_{F,EA}^{(t)}\} \\
m_5: & acc_{F,EA}^{(t)} = \frac{F(best_{EA}^{(t)}) - Min_F^{(t)}}{Max_F^{(t)} - Min_F^{(t)}} & m_{10}: & ARR = \frac{1}{m}\sum_{i=1}^{m} \frac{\sum_{j=1}^{p(i)}[f_{best}(i,j) - f_{best}(i,1)]}{p(i)[f^*(i) - f_{best}(i,1)]}
\end{array}
$$

along with *peak cover*, measuring the number of peaks found at moment t by having an individual within a peak's catchment area. In the experiments, catchment area has been simplified to a d-sphere. In addition, this paper introduces *peak cover error* (m_6) by analogy with *avg error of best individual* (m_7), and *exploration* (m_4) by analogy with *moment-of-inertia* (m_3) also considering quality of individuals.

3.2 Experiments

Variants of common benchmarks for DOPs were used in the experiments like the Moving Peaks Benchmark (MPB) [2], denoted by M_1. This paper also introduces dynamization of the Rosenbrock function's generalization, denoted by R_1:

$$f_{ros}(\boldsymbol{x}) = \sum_{i=1}^{d-1}\left((1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2\right). \tag{4}$$

For each pair of dimensions (i, j), where $1 \leq i < j \leq d$, rotation angle θ_{ij} is drawn from a normal distribution $\mathcal{N}(0, 0.5)$ to construct a rotation matrix $R_{ij}(\theta_{ij})$. Also, translation vector $(s_1, \ldots, s_d)^\intercal$ is drawn from $\mathcal{N}(0, 1)$ to construct a translation matrix. The final transformation matrix $M(t)$ is a product of all $\binom{d}{2}$ rotation matrices with translation as $M(t) = R_{12}R_{13}\cdot\ldots\cdot R_{(d-1)d}T$. Then, the objective function is $R_1(\boldsymbol{x}, \phi, t) = f_{ros}(\vec{M}(t)\cdot\boldsymbol{x})$.

To hinder solving problems M_1 and R_1, their variants have been introduced as M_2 and R_2 with applied cosine noise similar to that of the Griewank function:

$$M_2^{(i)}(\boldsymbol{x}, \phi, t) = M_1^{(i)}(\boldsymbol{x}, \phi, t) - w \prod_{i=1}^{n} \cos\left(\frac{c_i x_i}{\sqrt{i}}\right), \tag{5}$$

with R_2 constructed similarly. To compare the raw performance on compositions of popular functions (Sphere, Rastrigin, Weierstrass, Griewank, Ackley), MPBIL_C has been also tested on IEEE CEC'2009 Competition on Dynamic Optimization [6] problems $F_2 - F_6$, dynamized using $T_1 - T_6$ change types. Revisions of the benchmark generator have also been present on other similar events.

On M_1, M_2, R_1 and R_2, MPBIL_C was compared with a multimodal adaptation of Separable NES (SNES) [10], similar to Algorithm 1, but using SNES $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ update rules. Tests on problems $F_2 - F_6$ with change types $T_1 - T_6$ were carried using implementation available online [1]. The results allow for a direct comparison with a range of other contesting algorithms, using measures proposed in the competition. However, it should be noted that the benchmarks nor the employed measures promote multimodality. For this reason, the limitation on the number of function evaluations (FES) between timeslices was increased.

3.3 Results

Objective functions were parametrized with 5 random seeds chosen as numbers $1, \ldots, 5$, and results averaged over 30 runs for each seed. Thus, objective functions were always deterministic with respect to a random seed, while the algorithm behaved randomly with each subsequent run. F (its dynamization T) parametrized with random seed i is denoted by $F^{(i)}$ ($T^{(i)}$).

All benchmarks featured similar parameters concerning the amount of involved computations: population size $N = 300$, sub-evolution generations $N_{gen} = 800 - 1800$. In each case, MPBIL_C featured $M = 10$ Gaussians, performing best in preliminary experiments for most of functions with considered population size. Every objective function underwent $T = 60$ timeslices (changes).

Results are presented in Table 1. In MPBs, MPBIL_C was able to maintain and track a fair number of optima, resulting in lower errors, though exploration of search space suffers from fast convergence of Gaussians. Significant performance was achieved in R_1 and R_2.

Table 2 presents results of computations. Even though those objective functions might not necessarily promote multimodality, obtained results are in most cases comparable with best-performing algorithms of the competition, like jDE [3]. However, MPBIL_C performed poorly on problem F_4.

4 Conclusions

This paper presents a multimodal estimation of distribution algorithm MPBIL_C, capable of solving dynamic optimization problems. The algorithm models problem at hand with a Gaussian mixture model and controls parameters of the

Table 1. Performance of MPBIL_C and multimodal modification of SNES on M_1, M_2, R_1 and R_2 under $m_1 - m_{10}$ measures. M_1, M_2 were run with $K = 10$ peaks, $d = 5$ dimensions, $(\alpha, \beta, \delta, \rho) = (0.4, 0.01, 0.0001, 0.15)$, Gaussians initialized with $\boldsymbol{\sigma}_{init} = 5$ and restarted with $(t_r, \sigma_r) = (800, 0.5)$. R_1, R_2 were run with $d = 10$, domain restricted to $[-30, 30]^d$, adaptation parametrized with $(\alpha, \beta, \delta, \rho) = (0.4, 0.01, 0.00001, 0.15)$, Gaussians initialized with $\boldsymbol{\sigma}_{init} = 0.6$ and restarted with $(t_r, \sigma_i) = (1000, 0.3)$. Cosine noise was parametrized with $(w, c_i) = (2.0, 1.0)$ in M_2 and $(w, c_i) = (20, 0.1)$ in R_2. In all cases positions of Gaussians were initialized with uniform distribution.

	MPBIL_C					SNES					MPBIL_C					SNES				
Measure	$M_1^{(1)}$	$M_1^{(2)}$	$M_1^{(3)}$	$M_1^{(4)}$	$M_1^{(5)}$	$M_1^{(1)}$	$M_1^{(2)}$	$M_1^{(3)}$	$M_1^{(4)}$	$M_1^{(5)}$	$R_1^{(1)}$	$R_1^{(2)}$	$R_1^{(3)}$	$R_1^{(4)}$	$R_1^{(5)}$	$R_1^{(1)}$	$R_1^{(2)}$	$R_1^{(3)}$	$R_1^{(4)}$	$R_1^{(5)}$
fbog	115	152.5	119.3	111.7	107.3	50.2	69.62	67.84	45.41	73.2	2.806	2.866	2.731	2.763	2.664	106.7	107.2	67.58	64.85	109.6
emo	84.4	45.51	4.587	41.86	48.5	89.53	56.32	17.16	64.65	145.5	10.44	11.76	122.4	118.3	131.3	442.2	214.1	173.1	178.5	231.1
eb	63.96	114.2	86.45	8.617	69.53	60.54	55.62	93.16	76.29	49.9	.0433	.043	.043	.0466	.0432	28.99	8.14	9.668	7.328	19.25
accur	.6603	.9213	.8609	.8585	.7415	.3225	.4175	.6713	.5716	.2091	1	1	1	1	1	.9989	.9989	.999	.9994	.9989
ln(inrt)	14.75	14.29	14.61	14.7	14.8	14.05	13	13.67	15.01	13.75	1.812	1.796	1.788	11.04	11.04	1.715	1.694	1.639	1.932	1.941
pk_cvr	2.5	4.146	4.121	2.92	2.285	.9988	.9987	.9998	.965	.9866	.6599	.6439	.6599	.6477	.6636	.0303	.0283	.0131	.0253	.0162
pk_err	52.74	33.16	18.89	25.54	42.6	51.87	39.8	48	54.11	85.2	7.778	9.518	1.275	8.723	9.932	107.6	108.7	68.12	65.51	11.21
stab	.6606	.9242	.8637	.8613	.7439	.3233	.4115	.6812	.5716	.218	1	1	1	1	1	.9989	.9989	.999	.9994	.9989
arr	.2978	.6736	.6404	.5729	.5199	.0422	.1242	.568	.2379	.1181	.855	.8553	.8597	.8573	.8566	.898	.9005	.896	.8992	.8957
ln(explr)	13.47	14.46	14.24	14.37	14.25	13.62	13.12	13.8	14.7	13.18	11.76	11.77	11.61	11.91	11.79	1.685	1.712	1.618	1.802	1.822

	MPBIL_C					SNES					MPBIL_C					SNES				
Measure	$M_2^{(1)}$	$M_2^{(2)}$	$M_2^{(3)}$	$M_2^{(4)}$	$M_2^{(5)}$	$M_2^{(1)}$	$M_2^{(2)}$	$M_2^{(3)}$	$M_2^{(4)}$	$M_2^{(5)}$	$R_2^{(1)}$	$R_2^{(2)}$	$R_2^{(3)}$	$R_2^{(4)}$	$R_2^{(5)}$	$R_2^{(1)}$	$R_2^{(2)}$	$R_2^{(3)}$	$R_2^{(4)}$	$R_2^{(5)}$
fbog	117.5	89.05	85.12	10.42	104.8	4.049	4.088	4.033	4.017	4.11	1.089	1.087	1.087	1.086	1.089	56.76	85.47	77.34	57.65	126.3
emo	37.15	15.2	15.48	18.04	21.41	113.8	82.52	83.75	8.878	98.39	111.8	105.1	115.3	116.4	111.8	252.8	226.2	196.6	193.8	306.5
eb	86.23	7.181	75.97	73.2	8.903	3.668	3.69	3.65	3.638	3.696	.985	.9856	.9839	.9803	.9852	1.316	1.206	9.277	9.321	2.391
accur	.8291	.9091	.949	.9687	.7876	.0303	.0429	.0459	.0413	.033	1	1	1	1	1	.999	.9992	.9992	.999	.9987
ln(inrt)	18.01	17.95	18.38	18.36	17.81	8.798	9.447	8.718	8.549	8.356	1.815	1.813	1.81	11.05	11.05	11.5	11.37	11.6	11.44	11.45
pk_cvr	2.962	2.298	1.576	2.182	2.698	0	0	0	0	0	1	1	1	1	1	.0318	.0226	.0259	.0292	.0251
pk_err	35.27	25.47	35.35	34.6	39.4	62	48.36	46.76	54.1	73.31	.0566	.0578	.063	.0576	.0561	57.4	86.68	78.32	58.28	127.6
stab	.8292	.9091	.949	.9687	.7876	.0303	.0429	.0459	.0413	.033	1	1	1	1	1	.999	.9992	.9992	.999	.9987
arr	.597	.7165	.7963	.7675	.7368	0	0	0	0	0	.8541	.8518	.8565	.8522	.8538	.9237	.9257	.9243	.9262	.9234
ln(explr)	13.68	13.41	13.52	13.67	13.68	8.648	8.983	8.605	8.516	8.444	1.271	1.29	1.056	1.565	1.496	1.917	1.884	1.999	1.951	1.966

mixture using PBIL_C-like rules. By iteratively adjusting parameters throughout sub-generations, modes of the mixture are expected to converge to optima, which would then be utilized at timeslice to track changes of the objective function.

To assess performance of the proposed algorithm, numerous benchmarks have been run on popular DOPs under different change functions, noise and random seeds. Multimodality has been verified through a set of measures promoting diversity and peak coverage. The algorithm has outperformed a multimodal modification of SNES on benchmarks M_1, M_2, R_1, R_2 and gave results comparable in most cases to those CEC'2009 winning algorithm jDE on problems $F_2 - F_6$. It should be noted, however, that as the wide range of possible DOPs is far from being covered by synthetic benchmarks, MPBIL_C favors multimodality which is expected to be found in real-world problems, and not necessarily in benchmarks. Employed measures revealed that multimodality has been achieved, proving suitability of Gaussian mixtures for solving DOPs.

Table 2. Performance of MPBIL_C on $F_2 - F_6$ objectives. $F_2 - F_6$ were run with $(\alpha, \beta, \delta, \rho) = (0.1, 0.05, 0.0001, 0.5)$ and $N_{gen} = 800$; problems F_2, F_5, F_6 with $(\alpha, \beta, \delta, \rho) = (0.4, 0.01, 0.0001, 0.15)$, $N_{gen} = 1800$ and $(t_r, \sigma_r) = (1000, 5)$. Results obtained for remaining random seeds $2, \dots, 5$ were comparable with those presented.

Problem	Errors	$T_1^{(1)}$	$T_2^{(1)}$	$T_3^{(1)}$	$T_4^{(1)}$	$T_5^{(1)}$	$T_6^{(1)}$	$T_1^{(2)}$	$T_2^{(2)}$	$T_3^{(2)}$	$T_4^{(2)}$
F_2	Avg_best	.0305	.0606	.2396	.2517	.21	.2296	.0549	.0483	.1861	.2995
	Avg_worst	83.71	90.28	66.68	396.8	90.26	90.25	87.73	9.29	7.384	396.8
	Avg_mean	38.25	33.35	33.99	230.7	43.88	43.92	39.07	37.82	41.29	229.5
	STD	43.8	44.44	28.35	180	47.46	47.86	45.67	51.11	5.691	18.41
F_3	Avg_best	50.03	47.21	62.59	60.87	49.87	53.87	153.6	192.5	489	424.2
	Avg_worst	671	694.4	688.6	669.5	685.5	674.7	713.3	73.27	724.4	725.5
	Avg_mean	268.8	267.7	368.2	506.8	278.8	264.8	646.1	653.5	651.6	625.4
	STD	240.8	237.4	254	200.2	242.7	230.3	79.55	8.264	36.06	61.98
F_4	Avg_best	444.2	408.6	443.4	208.1	424.3	421.2	382.6	375.8	373.1	198.8
	Avg_worst	542.1	555.3	523.6	682.1	534.2	532.8	75.15	761.1	689.6	635.2
	Avg_mean	492.7	486.8	482.0	393.6	478.9	479.2	537.9	536.7	489.4	388.5
	STD	23.5	33.52	18.39	108.2	27.49	26.64	61.66	7.741	122.8	137.4
F_5	Avg_best	.1047	.1237	.1058	.3768	.2857	.3146	.1047	.1122	.1218	.4221
	Avg_worst	31.83	44.92	35.55	28.93	20.12	19.53	34.59	35.96	48.5	31.09
	Avg_mean	3.471	3.8	4.1493	4.195	2.471	1.999	2.682	2.075	1.027	4.735
	STD	5.971	35.5	8.7358	1.68	4.833	2.157	6.2	8.532	143.6	2.177
F_6	Avg_best	.1034	.1021	.1051	.1916	.192	.2021	.0992	.0977	.3466	.1913
	Avg_worst	42.09	8.692	4.355	424.6	67.56	69.22	64.09	87.78	73.47	361.8
	Avg_mean	14.21	17.02	6.943	24.23	21.45	21.62	2.008	2.088	29.15	23.81
	STD	152.4	42.16	23.58	73.3	42.07	41.73	44.27	45.8	41.43	57.88

References

1. Special session & competition on "Evolutionary computation in dynamic and uncertain environments". http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC-09-Dynamic-Opt/CEC09-Dyn-Opt.htm, accessed: 2014-5-1
2. Branke, J.: Memory enhanced evolutionary algorithms for changing optimization problems. In: Congress on Evolutionary Computation CEC99. pp. 1875–1882. IEEE (1999)
3. Brest, J., Zamuda, A., Boskovic, B., Maucec, M.S., Zumer, V.: Dynamic optimization using self-adaptive differential evolution. In: IEEE Congress on Evolutionary Computation. pp. 415–422. IEEE (2009)
4. Chen, S.H., Chen, M.C., Chang, P.C., Zhang, Q., Chen, Y.M.: Guidelines for developing effective estimation of distribution algorithms in solving single machine scheduling problems. Expert Syst. Appl. 37(9), 6441–6451 (Sep 2010)
5. Dong, W., Yao, X.: Unified eigen analysis on multivariate gaussian based estimation of distribution algorithms. Inf. Sci. 178(15), 3000–3023 (Aug 2008)
6. Li, C., Yang, S., Nguyen, T.T., Yu, E.L., Yao, X., Jin, Y., Beyer, H.G., Suganthan, P.N.: Benchmark generator for CEC 2009 competition on dynamic optimization. University of Leicester, University of Birmingham, Nanyang Technological University, Tech. Rep (2008)
7. Nguyen, T.T., Yang, S., Branke, J.: Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evolutionary Computation 6, 1–24 (Oct 2012)
8. Pelikan, M., Goldberg, D.E.: Hierarchical problem solving by the bayesian optimization algorithm. In: Proceedings of GECCO 2000. pp. 267–274. Morgan Kaufmann (2000)
9. Santana, R., Larraaga, P., Lozano, J.A.: Side chain placement using estimation of distribution algorithms. Artificial Intelligence in Medicine 39(1), 49–63 (2007)
10. Schaul, T., Glasmachers, T., Schmidhuber, J.: High dimensions and heavy tails for natural evolution strategies. In: GECCO. pp. 845–852. ACM (2011)
11. Sebag, M., Ducoulombier, A.: Extending population-based incremental learning to continuous search spaces. In: Parallel Problem Solving from NaturePPSN V. p. 418427. Springer (1998)
12. Shim, V.A., Tan, K.C., Cheong, C.Y., Chia, J.Y.: Enhancing the scalability of multi-objective optimization via restricted boltzmann machine-based estimation of distribution algorithm. Inf. Sci. 248, 191–213 (2013)
13. Yan, W., Xiaoxiong, L.: An improved univariate marginal distribution algorithm for dynamic optimization problem. AASRI Procedia 1(0), 166 – 170 (2012), AASRI Conference on Computational Intelligence and Bioinformatics
14. Yang, S.: Evolutionary computation for dynamic optimization problems. In: GECCO (Companion). pp. 667–682 (2013)
15. Yang, S., Yao, X.: Experimental study on population-based incremental learning algorithms for dynamic optimization problems. Soft Comput. 9(11), 815–834 (2005)
16. Yuan, B.: On the importance of diversity maintenance in estimation of distribution algorithms. In: Proceedings of GECCO 2005. pp. 719–726. ACM Press (2005)
17. Yuan, B., Orlowska, M.E., Sadiq, S.W.: Extending a class of continuous estimation of distribution algorithms to dynamic problems. Optimization Letters 2(3), 433–443 (2008)
18. Zhong, X., Li, W.: A decision-tree-based multi-objective estimation of distribution algorithm. 2013 Ninth International Conference on Computational Intelligence and Security 0, 114–11/8 (2007)