A Probabilistic Memetic Framework

Quang Huy Nguyen, Yew-Soon Ong, and Meng Hiot Lim
IEEE Trans. Evol. Comput., Vol. 13, No. 3, June 2009

Presented by Drew Buck
October 12, 2011

Outline

- Objective
  - What is a memetic algorithm?
  - How can it be improved?
- Design of the PrMF and APrMF
- Experiments
- Conclusions

Objective

What is a memetic algorithm and how can we improve it?

Optimization

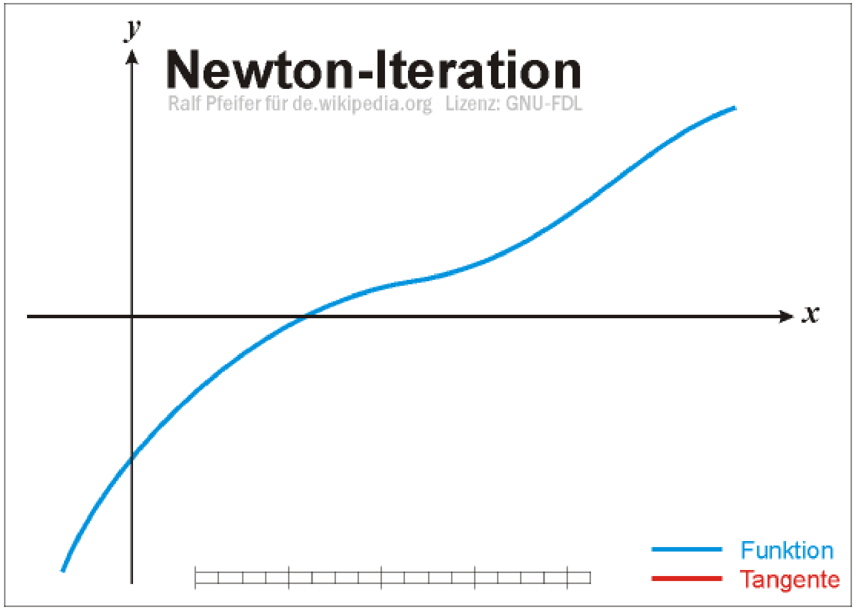
- Optimization is a process for seeking the best solution to a problem from a set of possible solutions.

- Deterministic Methods
  - Steepest Descent, Conjugate Gradient, Quadratic Programming, Linear Approximation…

- Stochastic Methods
  - Simulated Annealing, Tabu Search, Evolutionary Algorithms…

Deterministic Local Strategies

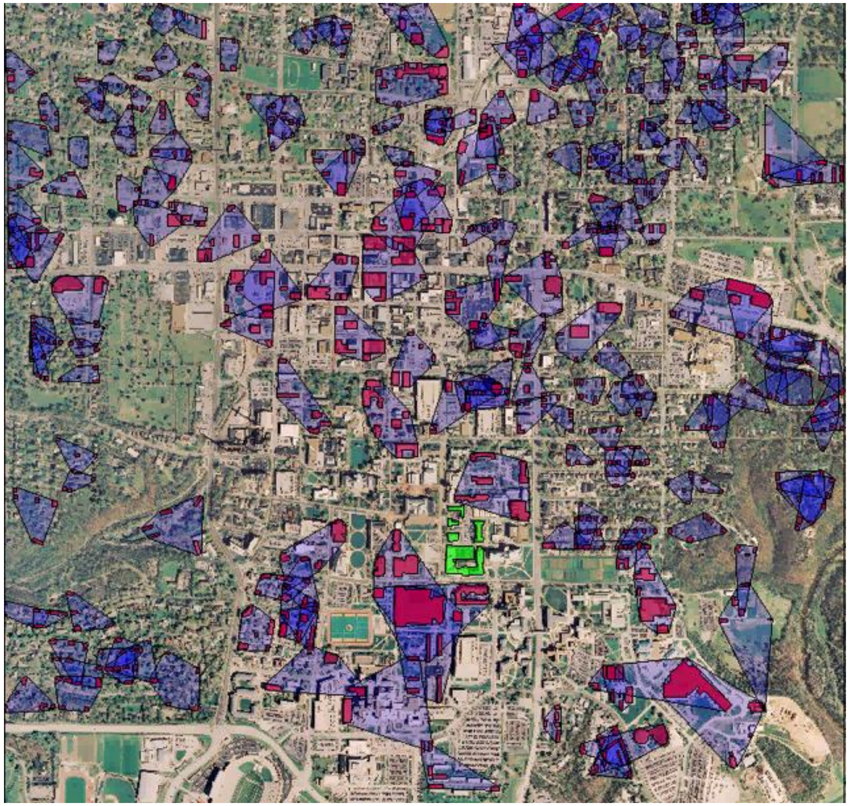
- Deterministic methods are usually efficient at finding local optima.
- They can be sensitive to initial starting conditions.
- They are more likely to stagnate on non-global optima than stochastic algorithms.



Example: Newton's Method (from Wikipedia)

Stochastic Algorithms

- Candidate solutions are drawn from a population.
- Good at solving non-convex, disjoint, or noisy solution spaces.
- Can take a long time to locate the exact local-optimum within the region of convergence.



Example: Object Set Matching

Memetic Algorithms

- Memetic algorithms are a recent extension of EAs which introduce individual learning as a separate process of local refinement for accelerating search.
- These hybrid algorithms combine the global search strategy of evolution with the local search strategy of deterministic methods.
- Recent studies have demonstrated that MAs converge to high-quality solutions more efficiently than conventional methods.

Two Individual Learning Strategies

- Several hybrid algorithms have been studied which seek to perform individual learning.
- Lamarckian learning
  - Each individual can modify its own genetic code during or after fitness evaluation to improve itself.
- Baldwin effect
  - The fitness of certain individuals is influenced by a local search without changing the genetic code.

Memetic Algorithm Outline

Procedure: Canonical Memetic Algorithm

Begin

/ Evolution - Stochastic Search Operators */*

Initialize: Generate an initial population;

While (Stopping conditions are not satisfied)

    Evaluate all individuals in the population.

    Select the subset of individuals, Ω_{il}, that should undergo the individual learning procedure.

    For each individual in Ω_{il}

        / Individual Learning –Local heuristics or Conventional exact methods */*

        - Perform individual learning using meme(s) with probability of P_{il} or frequency f_{il} for a t_{il} period.
        - Proceed with Lamarckian or Baldwinian learning.

    End For

    Generate a new population using stochastic search operators.

End while

End

Fig. 1. Outline of a memetic algorithm.

Design Considerations

- What is an appropriate search frequency f_{il} or probability P_{il} for applying local learning to an individual?
- To which subset of the population Ω_{il} should the local learning be applied?
- How long t_{il} should the local learning be run?
- Which local improvement procedure or meme is to be used?

PrMF and APrMF

- Unless one has *a priori* knowledge of a problem, it may be difficult to choose these parameters.
- Poor parameter choices may cause the MA to perform worse than evolution or individual learning alone.
- This paper presents a probabilistic and approximate probabilistic memetic framework (PrMF and APrMF) which governs at runtime whether evolution or individual learning should be favored.

Design of the PrMF and APrMF

Nonlinear Programming

- A target function $f(\mathbf{x})$ to be minimized
- A set of real variables, $\mathbf{x} \in \mathbb{R}^{ndim}, \; \mathbf{x}_{low} \leq \mathbf{x} \leq \mathbf{x}_{up}$
- A set of equality/inequality constraints, $g_w(\mathbf{x})$

- The goal is to locate the global minimum, $\mathbf{x}^*$ such that $\mathbf{x}^* = \text{argmin}_{\mathbf{x}} f(\mathbf{x})$ without violating the imposed constraints.

Types of Points

- The optimization problem is solved if at least one solution $\mathbf{x}'$ satisfies $f(\mathbf{x}') \leq f(\mathbf{x}^*) + \varepsilon$.

- A Type I point satisfies the above inequality.

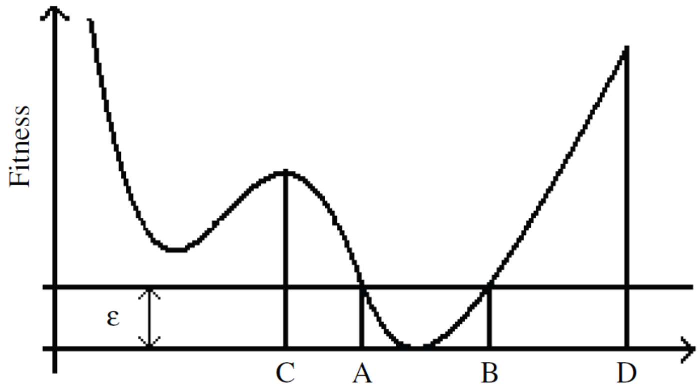


Fig. 2. Illustrations of type I and type II points.

- A Type II point lies in the basin of attraction containing Type I points.

- $p_1^{(k)}$ or $p_2^{(k)}$ is the probability that an individual in the population at generation k is a Type I or Type II point, respectively.

Probability of Points

Probability of finding at least one Type I point as a result of individual learning

$$P_l = 1 - \left(1 - p_2^{(k)}\right)^{f_{il} * n}$$

Total computational cost of individual learning

$$C_{il} = t_{il} \times \left(f_{il} \times n\right)$$

Number of generations that one may replace individual learning with evolutionary search for the same computational budget

$$\Delta_g = \frac{t_{il} \times \left(f_{il} \times n\right)}{t_{gs}}$$

Probability of having at least one Type I point as a result of stochastic evolution

$$P_g = 1 - \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n$$

n = population size
t_{il} = computational cost of individual learning
f_{il} = frequency of individual learning
t_{gs} = computational cost per generation of evolution

Deriving the Upper Bound of t_{il}

Use individual learning if it has a higher probability of reaching a Type I point over stochastic evolution.

$$P_l \geq P_g$$

$$\Leftrightarrow 1 - \left(1 - p_2^{(k)}\right)^{f_{il}*n} \geq 1 - \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n$$

$$\Leftrightarrow \left(1 - p_2^{(k)}\right)^{f_{il}*n} \leq \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n$$

Assuming that the global search method directs the search toward Type I points

$$\prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n \leq (1 - p_1^{(k)})^{n*\Delta_g}$$

Combining the above expressions

$$\left(1 - p_2^{(k)}\right)^{f_{il}*n} \leq \prod_{i=1}^{\Delta_g} \left(1 - p_1^{(k+i)}\right)^n \leq \left(1 - p_1^{(k)}\right)^{n*\Delta_g}$$

$$\Leftrightarrow \left(1 - p_2^{(k)}\right)^{f_{il}} \leq \left(1 - p_1^{(k)}\right)^{\Delta_g}$$

Deriving the Upper Bound of t_{il}

From before $$\left(1 - p_2^{(k)}\right)^{f_{il}} \le \left(1 - p_1^{(k)}\right)^{\Delta_g}$$

Take the logarithm of both sides $$f_{il} \ln\left(1 - p_2^{(k)}\right) \le \Delta_g \ln\left(1 - p_1^{(k)}\right)$$

$$\Leftrightarrow f_{il} \ln\left(1 - p_2^{(k)}\right) \le \frac{t_{il} f_{il} n}{t_{gs}} \ln\left(1 - p_1^{(k)}\right)$$

$$\Leftrightarrow \ln\left(1 - p_2^{(k)}\right) \le \frac{t_{il} n}{t_{gs}} \ln\left(1 - p_1^{(k)}\right)$$

Since $\ln\left(1 - p_1\right) < 0$, the above expression becomes

$$t_{il} \le \frac{t_{gs}}{n} \frac{\ln\left(1 - p_2^{(k)}\right)}{\ln\left(1 - p_1^{(k)}\right)}$$ or $$t_{il}^{\text{upper}} = \frac{t_{gs}}{n} \frac{\ln\left(1 - p_2^{(k)}\right)}{\ln\left(1 - p_1^{(k)}\right)}$$

PrMF

Procedure: PrMF

Begin

/ Evolution - Stochastic Search Operators */*

Initialize: Generate an initial population;

While (Stopping conditions are not satisfied)

Evaluate all individuals in the population.

For each individual in new population

/ Individual learning with t_{il} defined by the estimated theoretical upper bound */*

- Estimate the theoretical individual learning intensity bound, $t_{il}^{\text{upper}} = [(t_{gs})/(n)][(\ln\left(1 - p_2^{(k)}\right))/(\ln\left(1 - p_1^{(k)}\right))]$
- Perform individual learning using the specified meme for t_{il}^{upper} evaluations
- Proceed with Lamarckian or Baldwinian learning

End For

Generate a new population using stochastic search operators.

End while

End

Fig. 3. Outline of probabilistic memetic framework.

Example: Unimodal Sphere Function

- All points are in the single basin of attraction.
- $p_1^{(k)} < 1$ and $p_2^{(k)} = 1$
- t_{il}^{upper} approaches infinity, implying that local search will do better than stochastic evolutionary operators.

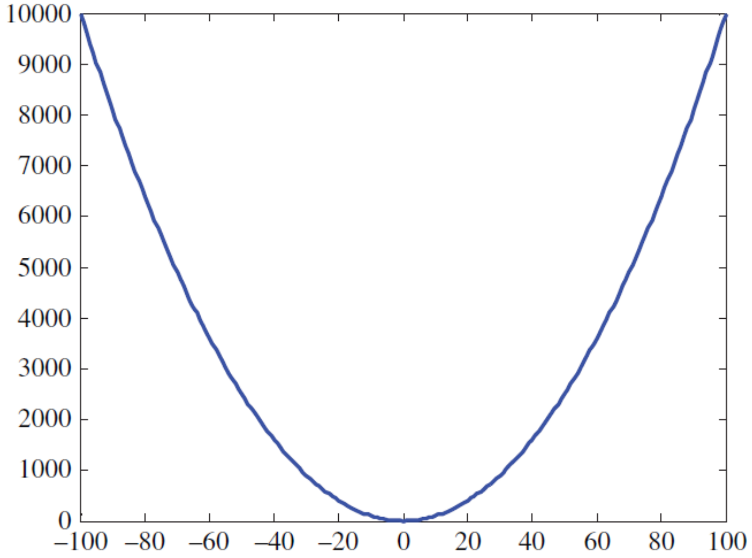


Fig. 4. Unimodal sphere function.

$$t_{il}^{\text{upper}} = \frac{t_{gs}}{n} \frac{\ln\left(1 - p_2^{(k)}\right)}{\ln\left(1 - p_1^{(k)}\right)}$$

Example: Multimodal Step Function

- The set of Type II points is the same as the set of Type I points.
- $p_1^{(k)} = p_2^{(k)}$
- $t_{il}^{\text{upper}} = \frac{t_{gs}}{n} \leq 1$
- Local search will not contribute to finding the global optimum.

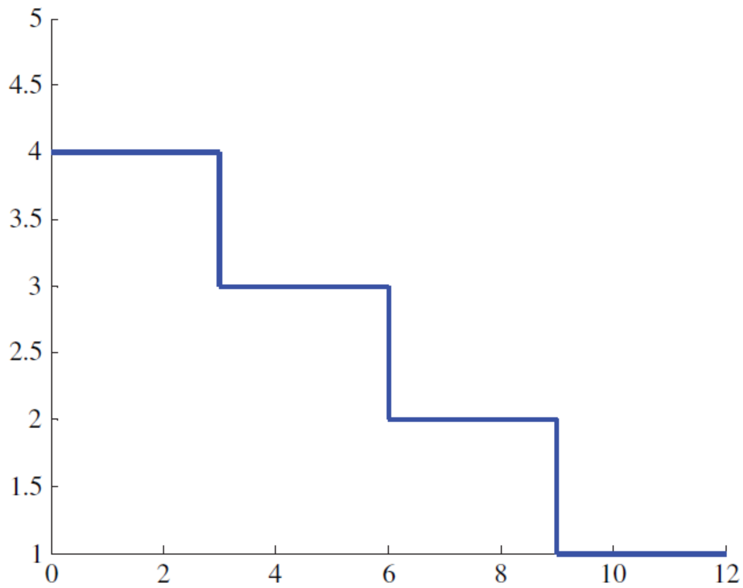


Fig. 5. Multimodal step function.

$$t_{il}^{\text{upper}} = \frac{t_{gs}}{n} \frac{\ln\left(1 - p_2^{(k)}\right)}{\ln\left(1 - p_1^{(k)}\right)}$$

APrMF

- $p_1^{(k)}$ or $p_2^{(k)}$ are not usually known and must be approximated.
- Introduce tracking capabilities on the search history and structure of each chromosome.
- Estimate *individual learning intensity* using search history of the entire population.

Procedure: APrMF
Begin

/ Start of Canonical MA */*
Initialize: Generate an initial population;
For the first few generations
    Evaluate all individuals in the population.
    For each individual $\mathbf{x}(i)$ in current population
        • Perform individual learning using the specified *meme with tracking capabilities* for a maximum of $t_{il}(i) = t_{il}^{\text{initial}}$ evaluations
        • Proceed with Lamarckian or Baldwinian learning
    End For
    Generate a new population using stochastic search operators.
End For

/ End of Canonical MA */*
While (Stopping conditions are not satisfied)
    Evaluate all individuals in the population
    For each individual $\mathbf{x}(i)$ in new population

        / Individual learning with adaptive t_{il} set according to upper bound and the expected value */*
            • Estimate learning intensity upper bound $t_{il}^{\text{upper}}(i)$ and expected learning intensity $t_{il}^{\text{expected}}(i)$ for individual $\mathbf{x}(i)$ using the *Individual Learning Intensity Estimation Scheme* outlined in Fig. 7.
            • **If** $(t_{il}^{\text{expected}}(i) \leq t_{il}^{\text{upper}}(i))$ **then**
            / increase budget for individual learning */*
            • $t_{il}(i) = f(t_{il}^{\text{expected}}(i))$
            • Perform individual learning using the specified meme for $t_{il}(i)$ evaluations
            • Proceed with Lamarckian or Baldwinian learning
        Else
            / Do not perform any individual learning */*
        End If

    End For
    Generate a new population using stochastic search operators.
  End while
End

Fig. 6. Outline of the approximate probabilistic memetic framework.

Individual Learning Intensity Estimation Scheme

Procedure: Individual Learning Intensity Estimation Scheme

Begin

/* *Estimate* p_1 *and* p_2 */

1. Identify set Ω, the q nearest chromosomes of $\mathbf{x}$ in database Φ.
2. Identify set Ω_{trace}, the q learning search traces associated with the q nearest chromosomes.
3. Find $\mathbf{x_{best}}$, the fittest individual in Ω_{trace}, i.e., $\mathbf{x_{best}} = \arg\min_{\mathbf{x}} \{ f(\mathbf{x}) | \mathbf{x} \in \Omega_{trace} \}$.
4. Find $\mathbf{x_1}$, the furthest ε -close point to $\mathbf{x_{best}}$, i.e., $\mathbf{x_1} = \arg\max_{\mathbf{x}} \{ \|\mathbf{x} - \mathbf{x_{best}}\| \, | \, f(\mathbf{x}) \le f(\mathbf{x_{best}}) + \varepsilon \}$
5. Estimate the upper bound for learning intensity,

$$t_{ls}^{\text{upper}} = \frac{t_{gs}}{n} \frac{\ln\left(1 - p_2^{(k)}\right)}{\ln\left(1 - p_1^{(k)}\right)} = \frac{t_{gs}}{n} \frac{\|\mathbf{x} - \mathbf{x_{best}}\|^{ndim}}{\|\mathbf{x_1} - \mathbf{x_{best}}\|^{ndim}}.$$

6. Estimate the expected value for learning intensity,

$$t_{il}^{\text{expected}} = \textit{average length of the search traces in } \Omega_{trace}.$$

End

Fig. 7. Outline of individual learning intensity estimation scheme.

Individual Learning Intensity Estimation Scheme

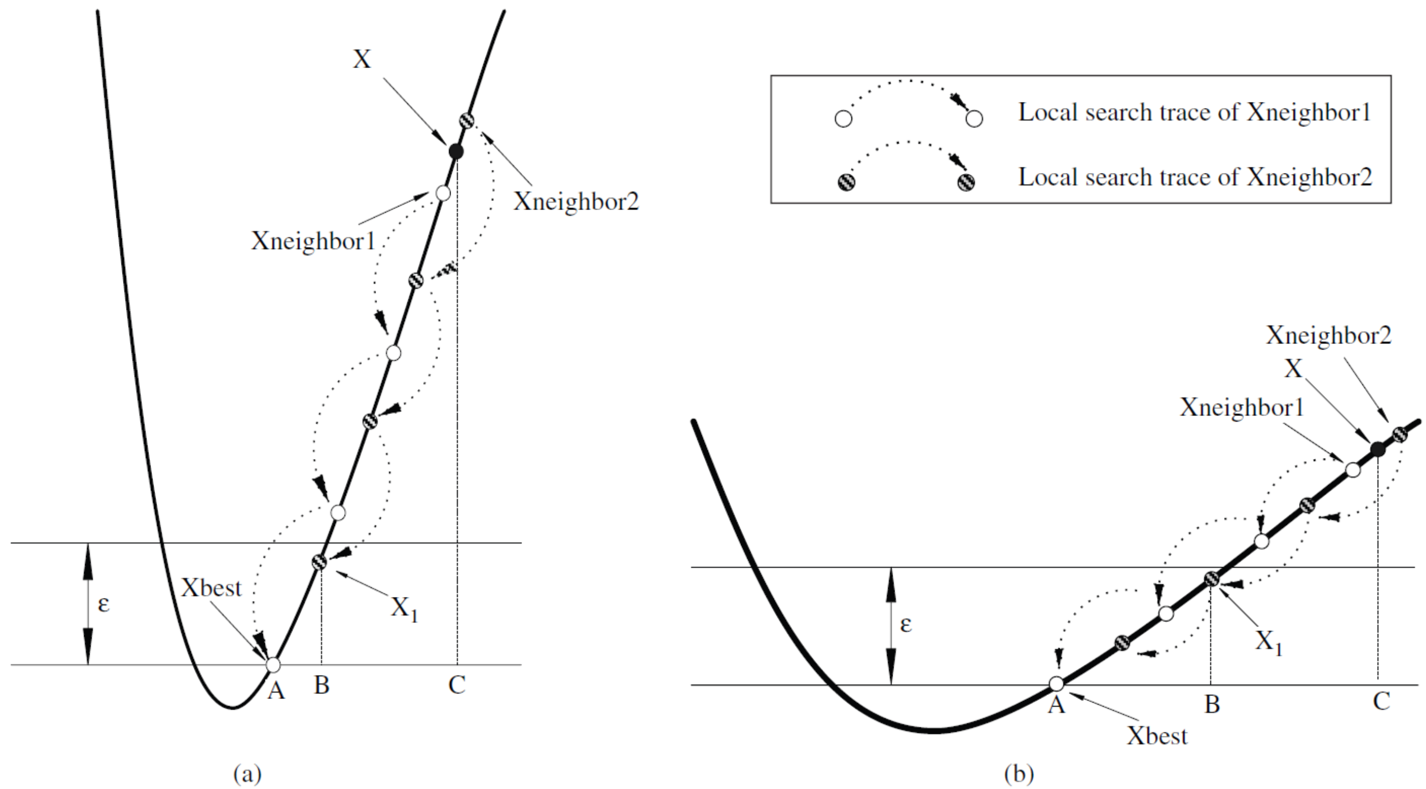


Fig. 8. Illustration of algorithm. (a) Case 1: narrow, deep basin. (b) Case 2: wide, shallow basin.

Case 1: $t_{il}^{upper} = \frac{CA}{BA} = 5$ Case 2: $t_{il}^{upper} = \frac{CA}{BA} = 2$ $t_{il}^{expected} = 3$

Experiments

Experiments

- Compared APrMF to the canonical MA analyzing the following:
  - Search quality and efficiency
  - Computational cost
  - Robustness
  - Simplicity and ease of implementation

Test Functions and Parameters

TABLE I
MULTIMODAL BENCHMARK FUNCTIONS USED IN THE STUDY (D IS THE NUMBER OF DIMENSIONS)

Benchmark test functions	Range of xi	Characteristics			Global optimum
		Epi*	Mul*	Disc*	
$F_{Sphere} = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	None	None	None	0.0
$F_{Step} = 6D + \sum_{i=1}^{D} \lfloor x_i \rfloor$	$[-5.12, 5.12]^D$	None	None	Medium	0.0
$F_{Griewank} = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-6, 6]^D$	Weak	High	None	0.0
$F_{Ackley} = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20$	$[-32, 32]^D$	None	Weak	None	0.0
$F_{Rastrigin} = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	$[-5, 5]^D$	None	High	None	0.0

TABLE II
PARAMETER SETTING OF APRMF

General parameters	
Global search	GA, DE and ES
Local search	DSCG, DFP and Simplex
Stopping criteria	100 000 evaluations or convergence to global optimum
Population size	50
Genetic algorithm parameters	
Encoding scheme	Real-coded
Selection scheme	Roulette wheel
Crossover operator	Two point crossover $p_c = 0.7$
Mutation operator	Gaussian mutation, $p_m = 0.03$
Differential evolution parameters	
Crossover probability	$p_c = 0.9$
ES parameters	
Selection method	$\mu + \lambda, \mu = 50, \lambda = 100$
Mutation operator	Gaussian mutation
Local search parameters	
Initial local search intensity $t_{il}^{initial}$	100 or 200 evaluations

Four configurations are examined:

1. APrMF with *initial* individual learning intensity configured to 100
2. APrMF with *initial* individual learning intensity configured to 200
3. Canonical MA with *fixed* individual learning intensity of 100
4. Canonical MA with *fixed* individual learning intensity of 200

Sphere and Step Functions

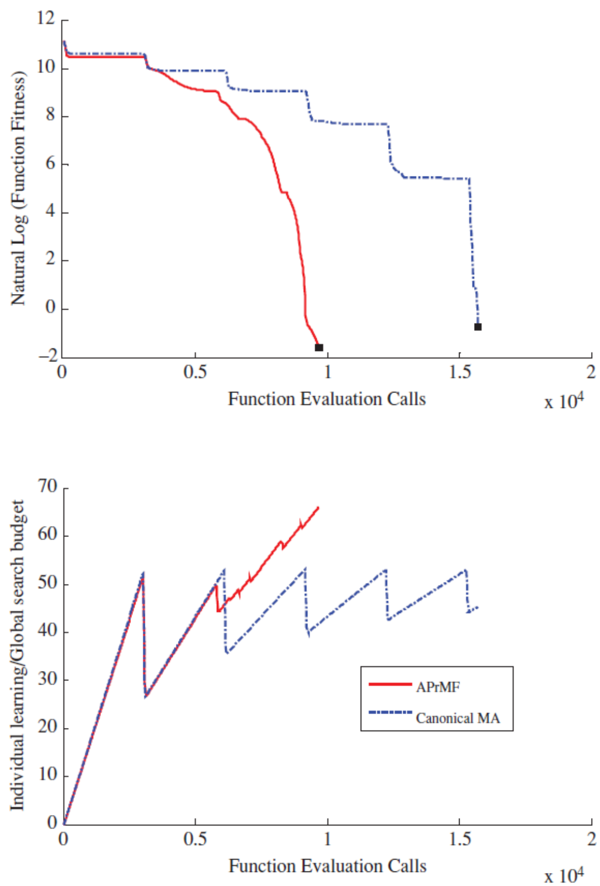


Fig. 9. Search trends of canonical MA (with a fixed individual learning intensity, $t_{il} = 100$) and APrMF (GA-DFP) on the unimodal Sphere function.

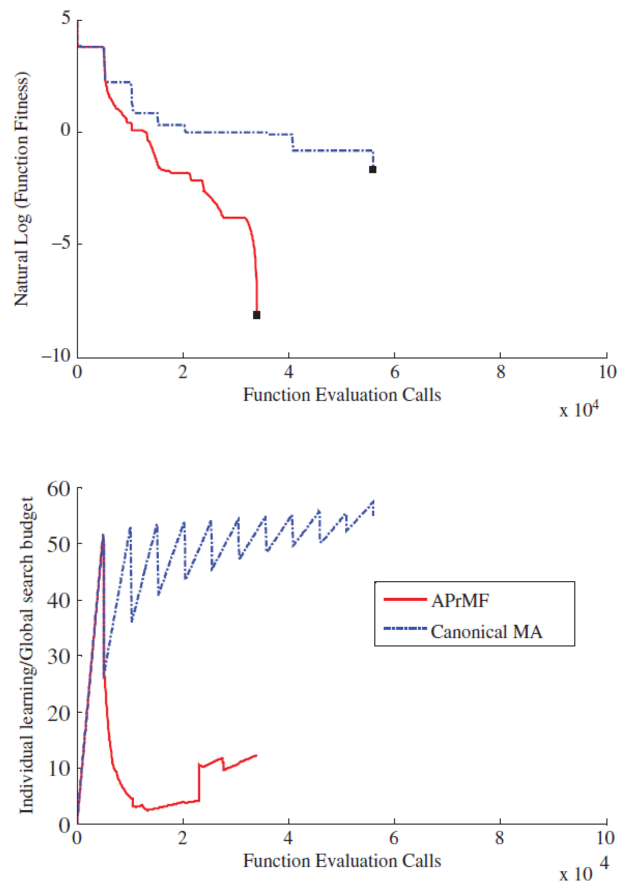


Fig. 10. Search trends of canonical MA (with a fixed individual learning intensity, $t_{il} = 100$) and APrMF (GA-DFP) on the Step function.

APrMF with Upward Trends

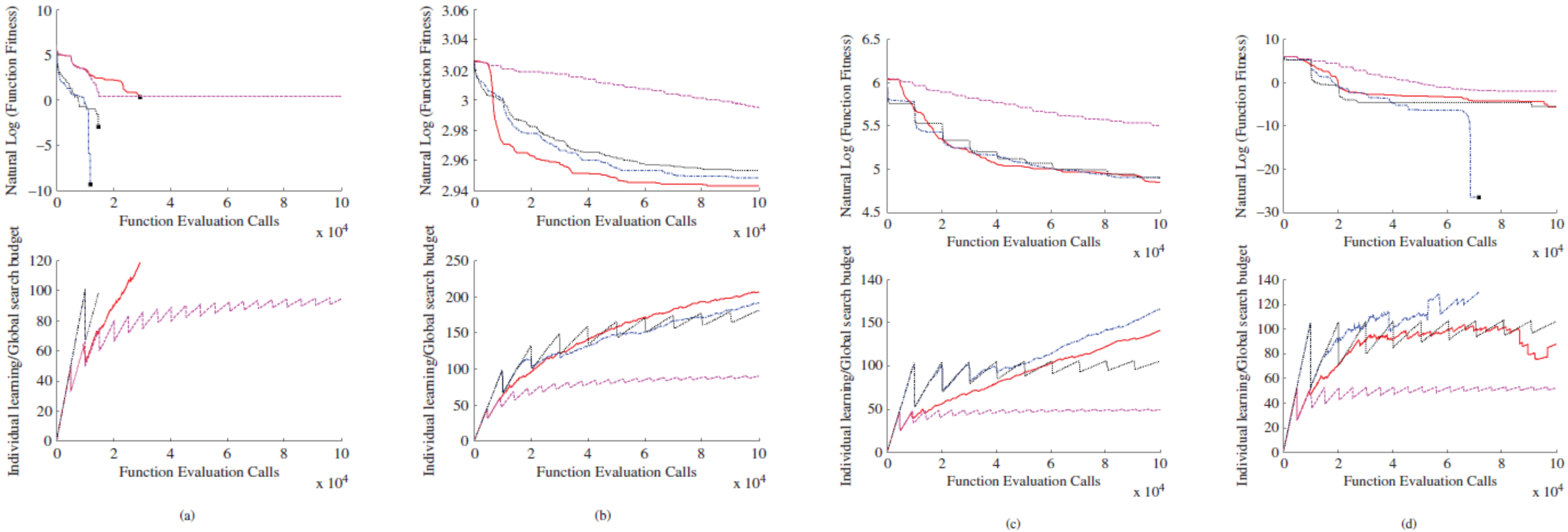
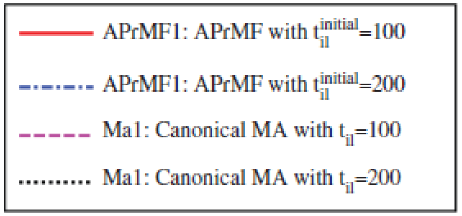


Fig. 11. APrMF with upward trends in individual learning/global search ratio and t_{il} [for legends, refer to Fig. 13(d)]. (a) Rastrigin DE-DSCG. (b) Ackley DE-Simp. (c) Rastrigin DE-DSCG. (d) Griewank GA-GSCG.



APrMF Adapting

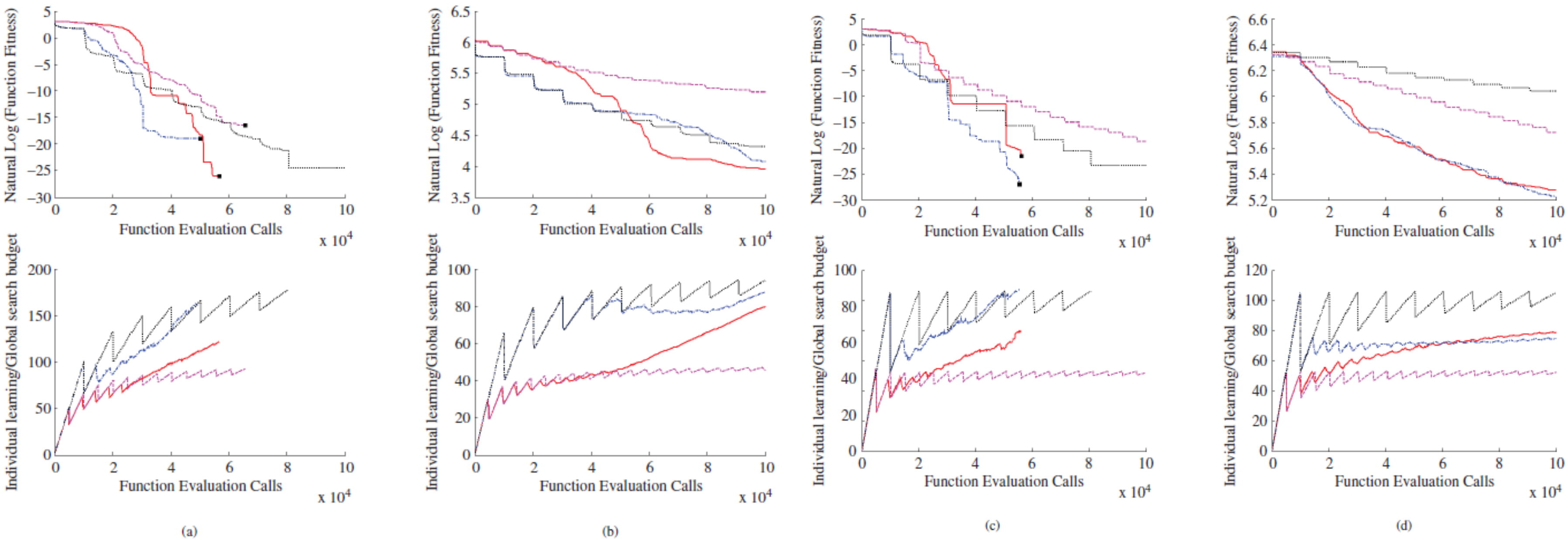
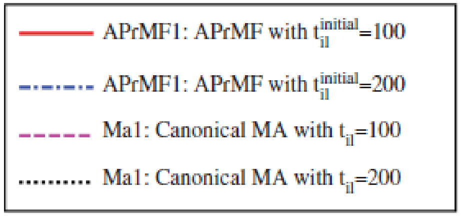


Fig. 12. APrMF Adapting the individual learning/global search ratio to its optimal configurationt_{il} [for legends, refer to Fig. 13(d)]. (a) Ackley De-DSCG. (b) Rastrigin ES-Simp. (c) Ackley GA-DSCG. (d) Griewank GA-Simp.



APrMF with Downward Trends

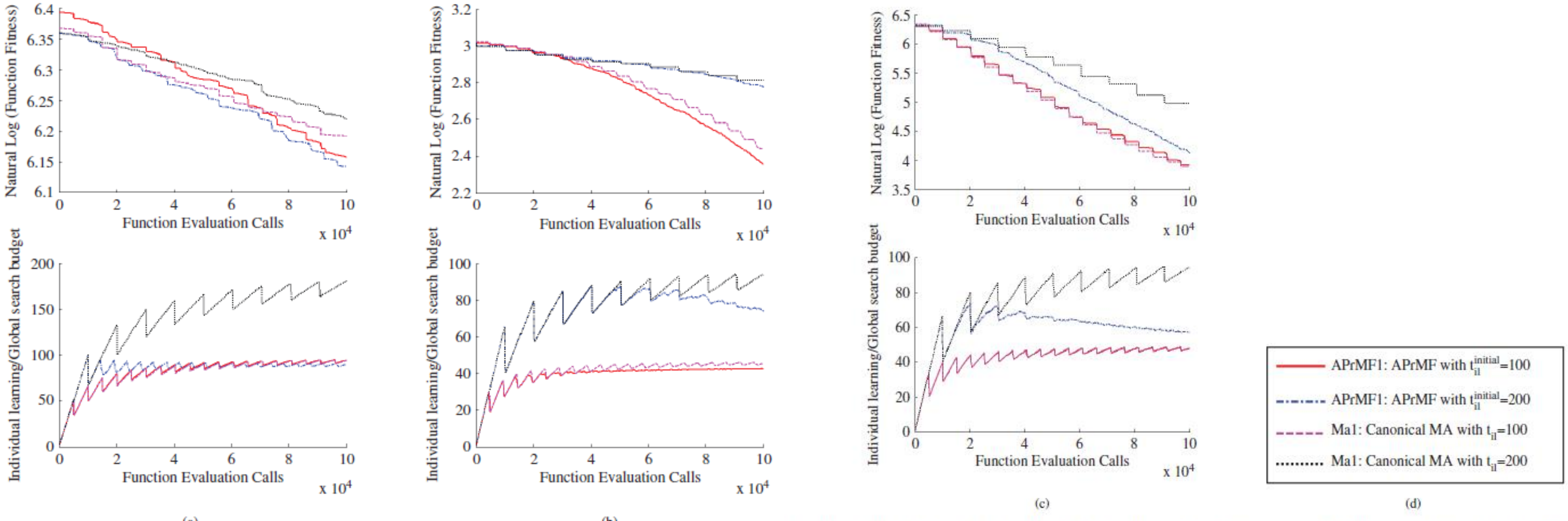


Fig. 13. APrMF with downward trends in individual learning/global search ratio and t_{il}. (a) Griewank DE-Simp. (b) Ackley ES-Simp. (c) Griewank ES-Simp. (d) Legend for Figs. 11–13.

Benchmark Comparison

TABLE III
BENCHMARK FUNCTIONS FOR REAL NUMBER OPTIMIZATION

Func	Benchmark test functions	Range	Characteristics		
			Epi*	Mul*	Disc*
1	$F_{Sphere} = \sum_{i=1}^{D} z_i^2$	$[-100, 100]^D$	None	None	None
2	$F_{Schwefel1.2} = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2$	$[-100, 100]^D$	High	None	None
3	$F_{Elliptic} = \sum_{i=1}^{D} \left(10^6\right)^{\frac{i-1}{D-1}} z_i^2$	$[-100, 100]^D$	None	None	None
4	$F_{Schwefel1.2+Noise} = \left(\sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2 \right) * (1 + 0.4\,\lvert N(0,1) \rvert)$	$[-100, 100]^D$	High	High	None
5	$F_{Schwefel2.6} = \max\{\lvert A_i x - A_i o \rvert\} \quad i = 1..D$	$[-100, 100]^D$	None	None	Medium
6	$F_{Rosenbrock} = \sum_{i=1}^{D-1} \left(100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2\right)$	$[-100, 100]^D$	High	High	None
7	$F_{Rastrigin} = \sum_{i=1}^{D} \left(z_i^2 - 10\cos(2\pi z_i) + 10\right)$	$[-5.12, 5.12]^D$	None	High	None
8	$F_{Griewank-R} = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1$	$[-\infty, +\infty]^D$	Weak	High	None
9	$F_{Ackley-R} = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20$	$[-32, 32]^D$	High	Weak	None
10	$F_{Rastrigin-R} = \sum_{i=1}^{D} (z_i^2 - 10\cos(2\pi z_i) + 10)$	$[-5.12, 5.12]^D$	High	High	None
11	$F_{Weierstrass-R} = \sum_{i=1}^{D} (\sum_{k=0}^{k\max} [a^k \cos(2\pi b^k (z_i + 0.5))]) - D \sum_{k=0}^{k\max} [a^k \cos(\pi b^k)]$	$[-0.5, 0.5]^D$	High	High	None
12	$F_{Schwefel2.13} = \sum (A_i - B_i x)^2$, $A_i = \sum_{j=1}^{D} (a_{ij} \sin\alpha_j + b_{ij}\cos\alpha_j)$, $B_i = \sum_{j=1}^{D} (a_{ij}\sin x_j + b_{ij}\cos x_j)$	$[-\pi, \pi]^D$	Weak	Weak	None
13	$F_{GrieRos} = \sum_{i=1}^{D} F_{Griewank-R}\left(F_{Rosenbrock-R}(z_i, z_{i+1})\right), z_{D+1} = z_1$	$[-5, 5]^D$	High	High	None
14	$F_{Scaffer} = \sum_{i=1}^{D} (F(z_i, z_{i+1})), z_{D+1} = z_1$, $F(x,y) = 0.5 + \frac{\sin^2\left(\sqrt{x^2+y^2}\right) - 0.5}{\left(1 + 0.001(x^2+y^2)\right)^2}$	$[-100, 100]^D$	Low	High	None
15	$F_{Hybrid1}$ (see f_{16} in [42])	$[-5, 5]^D$	High	High	Medium
16	$F_{Hybrid2}$(see f_{19} in [42])	$[-5, 5]^D$	High	High	Medium

TABLE IV
PARAMETERS CONFIGURATION OF APRMF BASED ON GA-DSCG

	10-D functions	**30-D functions**
Stopping criteria	100 000 evaluations	300 000 evaluations
Global search	Genetic algorithm	
Local search	DSCG	
Population size	50	
Encoding scheme	Real-coded	
Selection scheme	Roulette wheel selection	
Crossover operator	One point crossover $p_c = 0.7$	
Mutation operator	Gaussian mutation $p_m = 0.03$	
Initial local search intensity $t_{ls}^{initial}$	100 evaluations	300 evaluations

TABLE VII
MEMETIC ALGORITHMS OR HYBRID EA-LOCAL SEARCH USED IN COMPARISON

Algorithm name	**Description**
BLX-GL50	Hybrid Real-Coded Genetic Algorithms [43]
BLX-MA	Real-coded memetic algorithm with adaptive local-search probability and local search length [44]
DMS-L-PSO	Dynamic multi-swarm particle swarm optimizer with local search [45]
EDA	Continuous Estimation of Distribution Algorithms [46]
DEshcSPX	Differential evolution with crossover-based local search [47]
G-CMAES	Restart CMA Evolution Strategy With Increasing Population Size [48]

10-D Benchmark Results

TABLE VIII

SUCCESS MEASURE OF THE ALGORITHMS IN SOLVING THE 10-D BENCHMARK FUNCTIONS. FOR INSTANCE, 0.6 (25) ON F_{SPHERE} IMPLIES THAT APRMF INCURRED AN AVERAGE OF (0.6*1000) FUNCTION EVALUATION CALLS ON 25 SUCCESSFUL INDEPENDENT RUNS. A '-' ENTRY IN THE TABLE IMPLIES THAT THE RESPECTIVE ALGORITHM FAILS TO CONVERGE TO THE GLOBAL OPTIMUM. BOLD ITALIC ALSO HIGHLIGHTS THE BEST SEARCH PERFORMANCE (BASED ON PAIR-WISE T-TEST BETWEEN EACH RESPECTIVE ALGORITHM PAIRS)

	F_{Sphere}	$F_{Schwefel1.2}$	$F_{Elliptic}$	$F_{Rosenbrock}$	$F_{Griewank-R}$	$F_{Rastrigin}$	$F_{Schwefel2.13}$
APrMF	***0.6(25)***	11.0(25)	***0.6(25)***	***10.5(25)***	***17.6(20)***	***1.0(25)***	***31.9(19)***
BLX-GL50	19.0(25)	41.04(25)	-	51.8(25)	20.8(9)	20.4(3)	51.59(13)
BLX-MA	12.0(25)	36.96(25)	-	-	-	69.8(18)	-
DMS-L-PSO	12.0(25)	12.0 (25)	11.7(25)	54.7(25)	94.8(4)	35.7(25)	54.1(19)
EDA	10.0(25)	11.0(25)	16.3(23)	68.2(25)	75.9(1)	-	35.2(10)
DEshcSPX	22.9(25)	34.7(25)	89.2(20)	50.2(23)	97.3(21)	89.7(5)	-
G-CMAES	1.61(25)	***2.38(25)***	6.5(25)	**10.8(25)**	-	4.67(25)	**32.7(19)**

TABLE IX

RESULT OF T-TEST WITH 95% CONFIDENCE LEVEL COMPARING STATISTICAL VALUES FOR APRMF AND THOSE OF THE OTHER ALGORITHMS IN SOLVING THE 10-D BENCHMARK FUNCTIONS (S+, S–, AND ≈ INDICATE THAT APRMF IS SIGNIFICANTLY BETTER, SIGNIFICANTLY WORSE, AND INDIFFERENT, RESPECTIVELY)

	F_{Sphere}	$F_{Schwefel1.2}$	$F_{Elliptic}$	$F_{Rosenbrock}$	$F_{Griewank-R}$	$F_{Rastrigin}$	$F_{Schwefel2.13}$
BLX-GL50	s+	s+	-	s+	s+	s+	s+
BLX-MA	s+	s+	-	-	-	s+	-
DMS-L-PSO	s+	s+	s+	s+	s+	s+	s+
EDA	s+	≈	s+	s+	s+	-	s+
DEshcSPX	s+	s+	s+	s+	s+	s+	-
G-CMAES	s+	s–	s+	≈	-	s+	≈

30-D Benchmark Results

TABLE X

SUCCESS MEASURE OF THE ALGORITHMS IN SOLVING THE 30-D BENCHMARK FUNCTIONS. A '-' ENTRY IN THE TABLE IMPLIES THAT THE RESPECTIVE ALGORITHM FAILS TO CONVERGE TO THE GLOBAL OPTIMUM. BOLD ITALIC ALSO HIGHLIGHTS THE BEST SEARCH PERFORMANCE (BASED ON PAIR-WISE T-TEST BETWEEN EACH RESPECTIVE ALGORITHM PAIRS)

	F_{Sphere}	$F_{Schwefel1.2}$	$F_{Elliptic}$	$F_{Griewank-R}$	$F_{Rastrigin}$
APrMF	***0.6(25)***	87.0(25)	***1.0(25)***	***25.5 (25)***	***5.9(25)***
BLX-GL50	58.05(25)	159.6(25)	-	66.3(25)	-
BLX-MA	32.13(25)	-	-	-	238.8(9)
DMS-L-PSO	5.13(25)	129.6(25)	285.3(21)	57.4(24)	-
EDA	150.1(25)	159.6(25)	219.3(25)	129.93(25)	-
DEshcSPX	89.4(25)	299.3(2)	-	148.1(21)	-
G-CMAES	4.5(25)	***13.0(25)***	42.7(25)	-	**6.1(25)**

TABLE XI

RESULT OF T-TEST WITH 95% CONFIDENCE LEVEL COMPARING STATISTICAL VALUES FOR APRMF AND THOSE OF THE OTHER ALGORITHMS IN SOLVING THE 30-D BENCHMARK FUNCTIONS (S+, S–, AND ≈ INDICATE THAT APRMF IS SIGNIFICANTLY BETTER, SIGNIFICANTLY WORSE, AND INDIFFERENT, RESPECTIVELY)

	F_{Sphere}	$F_{Schwefel1.2}$	$F_{Elliptic}$	$F_{Griewank-R}$	$F_{Rastrigin}$
BLX-GL50	s+	s+	-	s+	-
BLX-MA	s+		-	-	s+
DMS-L-PSO	s+	s+	s+	s+	-
EDA	s+	s+	s+	s+	-
DEshcSPX	s+	s+	-	s+	-
G-CMAES	s+	s–	s+	-	

Conclusions

- We can improve the canonical MA by deciding at runtime whether to use evolution or individual learning.
- APrMF outperforms other comparable methods on common benchmark functions.
- APrMF reduces the number of free parameters that must be set by the user.