Chapter 14
Memetic Algorithms in the Presence of Uncertainties

Yoel Tenne

14.1 Motivation

Memetic Algorithms have proven to be potent optimization frameworks which are capable of handling a wide range of problems. Stemming from the long-standing understating in the optimization community that no single algorithm can effectively accomplish global optimization [940], memetic algorithms combine global and local search components to balance exploration and exploitation [368, 765]: the global search explores the function landscape while the local search refines solutions. In literature the terms memetic algorithms [615, 673] and hybrid algorithms [325] refer to the same global–local framework just described. The merits of memetic algorithms have been demonstrated in numerous publications, [374, 375, 686, 688].

However, while optimization algorithms are often conceived and tested on synthetic mathematical problems, real-world applications can be significantly different. One such major difference is that real-world problems often induce uncertainty in the optimization problem and studies identify four common scenarios [425]:

1. a model approximates the objective function and provides the optimizer with predicted objective values having an unknown error
2. the variables can stochastically fluctuate and it is required to find a solution which is insensitive to these fluctuations
3. the responses from the objective function are corrupted by noise and
4. the problem (objective function, constraints) is dynamic, that is, varies with time.

As such, baseline memetic algorithms developed using synthetic problems can perform poorly in such uncertain settings and this has motivated research into new

Yoel Tenne
Department of Mechanical Engineering and Science-Faculty of Engineering,
Kyoto University, Japan
e-mail: `yoel.tenne@ky3.ecs.kyoto-u.ac.jp`

and dedicated memetic frameworks. As such the goal of this chapter is to survey representative studies on memetic algorithms in the four uncertainty classes. In the remainder of the chapter we consider without loss of generality the minimization problem

$$\begin{aligned} &\min\ f(x) \\ &\text{s.t.}\ \ g_i(x) \leqslant 0\,, i = 1 \ldots k \end{aligned} \tag{14.1}$$

as the baseline optimization problem.

The remainder of this chapter is as follows: Section 14.2 surveys Algorithms for optimization with uncertainty due to approximation, Section 14.3 deals with Algorithms for robust optimization, Section 14.4 surveys Algorithms for noisy optimization problems, Section 14.5 deals with dynamic optimization problems and lastly Section 14.6 concludes the chapter.

14.2 Uncertainty Due to Approximation

Current research in engineering and science often replaces real-world laboratory experiments with analysis-codes, that is, computationally-intensive simulations which model real-world physics with high accuracy [881]. The approach allows to reduce the cost and duration of the design process and is being widely used, for example in aerospace [307, 725] electrical engineering [484] and chemistry [603]. Such computer simulations are typically *computationally expensive*, that is, each simulation call requires minutes to hours of CPU time. This makes many optimization algorithms, and particularly computational intelligence ones (such as evolutionary algorithms, particle swarm optimizers and so on) inapplicable since they require many thousands of function evaluations making the optimization process prohibitively expensive.

There are two main approaches to combat this difficulty. First, parallelization allows to reduce the wall-clock time [192, 725]. While the approach can be efficient one potential obstacle is that for commercial analysis-codes there is typically a licence restricting the number of concurrent simulations which can be run.

A complementary approach is that of modelling. Based on the ‘plug-in’ concept in statistics, the idea is to create a computationally cheaper mathematical approximation of the expensive simulation and to use it instead during the optimization search. The optimization algorithm then obtains the (predicted) objective values from the model in a fraction of the time when compared to using the true (expensive) simulation [283, 603]. Representative model types include:

- Quadratics [646]: the simplest models which capture function curvature and have the general form

$$\mathscr{S}(x) = \frac{1}{2}x^{\mathrm{T}}Hx + x^{\mathrm{T}}g + c \tag{14.2}$$

  where coefficients are typically determined by a least-squares fit.

- Radial Basis functions (RBFs) [85]: the model is defined as a linear combination of kernel basis functions

$$\mathscr{S}(x) = \sum_{i=1}^{k} \alpha_i \phi(\|x - x_i\|) \tag{14.3}$$

where α_i is a scalar coefficient and x_i is an interpolation point. The coefficients are obtained from the Lagrangian interpolation condition

$$\Phi \alpha = f \tag{14.4}$$

where Φ is the interpolation matrix ($\Phi_{i,j} = \phi(\|x_i - x_j\|)$) and f is the vector of responses ($f_i = f(x_i)$).
- Kriging [172]: a statistically-oriented approach which models the function as a combination of a global 'drift' function (typically a constant β) and a stochastic function $Z(x)$ providing local adjustments so the model becomes

$$\mathscr{S}(x) = \beta + Z(x)\,. \tag{14.5}$$

The stochastic function is a Gaussian process with a zero mean and variance σ. Model parameters are typically calibrated by maximum-likelihood to best fit the data [554].
- Artificial Neural Networks (ANN) [68]: a biologically-inspired approach which uses an array of inter-connected 'neurons' (processing units). The ANN is trained using available data and learns the input-output mapping.

While models alleviate the bottleneck of high computational cost they introduce uncertainty into the optimization problem: the optimizer now needs to operate based on the responses of the model but those are inherently inaccurate as the model is trained using a typically small sample (since evaluations are expensive). The extent of inaccuracy is unknown and depends on various factors such as the dimension and landscape complexity of the objective function and the sample size [544, 851].

Model inaccuracy implies that the optimizer is searching on a deformed landscape with uncertainty regarding its 'goodness'. If the model accuracy is poor then the optimizer may even converge to a false optimum (an optimum of the model which is not an optimum of the true expensive function) [426]. This implies that to be effective model-assisted frameworks must account for this *uncertainty due to approximation* and several approaches have been proposed.

In [307, 439] the authors proposed the Inexact Pre-Evaluation (IPE) framework which uses the expensive function in the first few generations (typically 2–3) and then uses the model almost exclusively while only a portion of the elites are evaluated with the expensive function and are used to update the model. The approach was later incorporated into a hierarchical distributed algorithm [803] which uses 'layers' of optimization, for example, at each layer an EA uses an analysis code of different fidelity. Promising individuals would then migrate from the computationally cheap low-fidelity layer to the expensive high-fidelity layer to obtain a more accurate fitness and vice versa. The idea was later expanded such that each layer may use different solvers, for example an EA and a gradient-based resulting in a memetic like framework [440]. By using the high-accuracy simulation and a gradient search the framework can diminish the effect of the low fidelity simulations.

In [426] the authors proposed the Controlled Evaluations (CE) framework which monitors the model accuracy using cross-validation: a cache of previously evaluated vectors is split into two disjoint sets and a model is trained using one set and tested on the complementary set. Model accuracy is then measured by the mean squared error (MSE)

$$MSE = \frac{1}{k}\sum_{i=1}^{k}\big(\mathscr{S}(x) - f(x)\big)^2 \tag{14.6}$$

for a test set of k vectors. The authors examined both individual-based control (at each generation evaluating a few vectors with the expensive function) and generational-based control (every few generations evaluating all individuals with the expensive function). A fuzzy logic rule adapted the frequency of expensive evaluations, that is, it increased the number of expensive evaluations when the MSE is too large and vice versa. A related memetic approach was proposed in [305] where for an expensive multiobjective optimization problem. The EA was used for a certain number of generations and then an ANN was trained to predict objective responses. The framework then used a gradient local search to refine solutions while monitoring the goodness of the ANN using (14.6).

The trust-region (TR) framework is another option for managing optimization with approximation uncertainty and has a long standing history in nonlinear programming (and unrelated to expensive black-box optimization). The idea is to perform a sequence of restricted steps around the optimum instead of a one-shot global optimization of the model. Starting from an initial guess $x^{(0)}$ then at each iteration $i = 0, 1, \ldots$ a model is trained and the framework performs a trial step, that is, it seeks an optimum of the model constrained to the trust-region ($\mathscr{T}$) where

$$\mathscr{T} = \{x : \|x - x^{(i)}\|_p \leqslant \Delta\}, \quad p = 1 \text{ or } 2, \tag{14.7}$$

where Δ is the TR radius. This defines the constrained optimization problem

$$\begin{aligned} \min \quad & \mathscr{S}(x) \\ \text{s.t.} \quad & x \in \mathscr{T} \end{aligned} \tag{14.8}$$

which gives a minimizer x_{m}. Next, the framework examines the success of the trial step with the merit value

$$\rho = \frac{f(x^{(i)}) - f(x_{\mathrm{m}})}{\mathscr{S}(x^{(i)}) - \mathscr{S}(x_{\mathrm{m}})}, \tag{14.9}$$

where $\rho > 0$ indicates the trial was successful, that is, the predicted optimum indeed improves on the current iterate ($\rho = 1$ indicates a perfect agreement between the model prediction and the true function). Based on the value of ρ the framework then updates the iterate and the TR, for example:

- if $\rho > 0$: centre the TR at x_{m} (so $x^{(i+1)} = x_{\mathrm{m}}$) and increase Δ .
- otherwise decrease Δ .

A merit of the TR framework is that it guarantees asymptotic convergence to an optimum of the true (expensive) objective function [141, 771] which has motivated using it in memetic settings.

Reference [78] seems to be among the first to propose a TR-based memetic framework. It used a variant of the pattern search algorithm as a global search which gradually restricted the search to zoom in on an optimum. In case no improvement was made over the current iterate the authors proposed invoking a gradient-free local search to refine solutions.

Later [681, 682] proposed memetic frameworks combining an EA as a global search where at each generation every non-duplicated vector in the population was refined using a TR local search with local RBF models. The extent of the memetic refinement was limited to k iterations (prescribed a-priori by the user). If the local search found an improved (true) solution after k iterations then another round was performed but otherwise it terminated and the resultant solution replaced its original in the population in a Lamarckian updating scheme.

In [878, 879] the authors proposed a TR memetic framework which uses quadratic models and clustering. An EA performs global exploration and it directly evaluates the expensive objective function. Every several generations the framework would cluster the population using the k-means algorithm [543] to identify if the population is converging around previously found optima. The idea is to improve the search by identifying basins of attractions (by clustering) and invoking the local search only from solutions considered to lie in yet unexplored basins [891]. The local search is based on the DFO algorithm which is a gradient-free TR local search algorithm [140, 141].

To further improve search efficiency and leverage on the power of models several studies have proposed using models both in the global and local search phases. For example, [963] extended the framework from [681]: an EA searches over a global Kriging model and a number of solutions were then refined using a TR local search with RBF models. After the local search the refined solutions replace the originals in the population in a Lamarckian update scheme. A related study [962] proposed a framework which uses a global Kriging model but with multiple local searches (possibly performed in parallel) where each is performed based on a different model type. The idea is that occasionally an inaccurate model can actually yield a fast improvement in the search [685] and so performing multiple searches and selecting the best solution among them can improve the search effectiveness (the study used a quadratic model and an RBF one during the local search). Continuing the multiple models approach, [522] has recently proposed a framework relying on ensembles of models as well as smoothing models. The framework uses an ensemble of different local models where the individual predictions by each model are aggregated into a single response based on the models' accuracy. The framework also employs a smoothing-model (low-order polynomial) to reduce the number of optima and simplify the landscape. During the search the framework chooses between the optimum predicted by the smoothing model and the ensemble. The authors have also presented a multiobjective variant of the framework.

Another development was that of model-adaptive frameworks [876, 879, 880]. The approach is motivated by the tenet that an optimal model is problem dependant but often there is insufficient a-priori information to select the optimal type [476, 557]. As such, a model-adaptive framework aims to autonomously select the best model from a family of candidates. To achieve this the framework leverages on a rigorous statistical model selection theory: it assesses the goodness of a model based on its maximum likelihood which is a statistical measure indicating how well a model fits the data [526, 718]. When comparing different candidate models the one having the highest likelihood is chosen as the best predictor of the data.

Leveraging on these ideas, [876] proposed a model-adaptive memetic framework which uses a DFO-like local search with Kriging models and selected at each iteration an optimal local model type. A follow-up study [880] then extended model-adaption to select an optimal global model as well. The proposed framework used an RBF neural network as a global model and selected an optimal RBF kernel for it out of the four candidate kernel functions based on the MSE criterion (14.6). Next, an EA would search for an optimum of the model and then a TR local search would improve the predicted optimum. The local search followed the classical TR procedure described earlier but with the addition of monitoring the number of points in the TR. If the trial step was unsuccessful and there were too few points in the TR a new point would be added to improve the model. Also, the framework selected an optimal model during the local search iterations. The global–local process would repeat until the optimization budget was exhausted. Algorithm 25 gives a pseudocode of the framework. Three variants of Ratle's algorithm [757] were used each with a different RBF model (multiquadric, linear and inverse multiquadric) where the model type was fixed throughout the search. The proposed framework showed statistically significant performance advantage over the three variants indicating the merit of model adaption. Lastly, the framework and Ratle's algorithm were also used in an airfoil shape optimization (an 11 dimensional problem) and again showed a statistically-significant performance advantage. Overall, performance analysis showed that adapting the model improves the optimization search.

14.3 Uncertainty Due to Robustness

In many real-world applications a system needs to operate under a range of conditions and not a single fixed one. For example, an engine should maintain efficiency over a range of operating speeds or an aircraft fleet assignment should maintain punctuality while accounting for a range of weather conditions and so on. In these cases and similar ones elements of the problem are not crisp but can stochastically assume any value within a known range. In such settings the optimization goal is typically not to find the best global optimum but rather a robust solution which yields a 'good' objective response and which is relatively insensitive to fluctuations.

Algorithm 25. A Global–Local Model-Adaptive Memetic Framework [880]

```
1  generate initial sample;
2  repeat
3      global search phase: select model type by maximum likelihood;
4      train global model;
5      locate model optimum with EA;
6      select starting point for local search;
7      local search phase: repeat
8          select model type by maximum likelihood;
9          train local model;
10         perform trial step;
11         update TR based on step, improve model if necessary;
12     until k iterations or convergence ;
13 until until evaluations budget exhausted ;
```

Robust optimization problems can be classified according to which elements of the problem vary:

- objective function (for example, noise in instruments measuring the objective values).
- variables (for example, manufacturing inaccuracies).
- operating conditions (for example, the ambient temperature in which a system operates).

As a side note, a solution which can be adapted to yield a high-quality response is termed flexible [837]. In contrast, a robust solution requires no adaption.

Given the stochastic nature of the variations, statistical decision theory [203, 610] suggests three main criteria for selecting robust solutions (for simplicity we consider an unconstrained minimization problem). The robust solution should provide a bound on the worst case performance, implying (in minimization) a min-max formulation, that is

$$\min\max f(x) . \tag{14.10}$$

The robust solution should minimize the expected objective value, mathematically

$$\min F(x) \tag{14.11}$$

where

$$F(x) = \int_{-\infty}^{+\infty} f(x+\delta)\, p(\delta) \tag{14.12}$$

and x is the baseline design vector (nominal settings), δ is a fluctuation and $p(\delta)$ is its probability density function. In practice both the distribution $p(\delta)$ and the effect of fluctuations on the objective response (or uncertainty propagation [229]) are unknown and so algorithms use Monte Carlo sampling [526] to generate the empirical unbiased estimate

$$\hat{F}(x) \simeq \frac{1}{N}\sum_{i=1}^{N} f(x+\delta)\,. \tag{14.13}$$

The robust solution should minimize both the expected objective response and its variance since (14.12), (14.13) can still yield a small expected value even when there are large positive and negative responses cancelling each other. This scenario also considers the objective variance

$$Var(f(x)) = \int_{-\infty}^{\infty} (f(x+\delta) - F(x))^2\, p(\delta)\,. \tag{14.14}$$

The problem formulation is then

$$\begin{aligned} &\min\ F(x) \\ &\min\ Var(f(x)) \end{aligned} \tag{14.15}$$

which is a bi-objective optimization problem. As before, when the exact information is unavailable algorithms use the empirical unbiased estimate of the variance

$$\widehat{Var}(F) = \frac{1}{k-1}\sum_{i=1}^{k}(f-\hat{F})^2 \tag{14.16}$$

In [869] the authors proposed a memetic algorithm for robust optimization of digital filters where the uncertainty in performance is due to material imperfections. The problem formulation involves both three parameters (which can assume a range of values) and 12 design variables defining the filter geometry (termed control factors). The goal of the optimization was to find a filter with a robust frequency response. Following the Taguchi method [870], the authors used a full-factorial design for the three parameters (defining ‘low’ and ‘high’ settings for each parameter and evaluating all 8 combinations). The authors then defined a sound-to-noise ratio (SNR) as a measure of robustness and maximized it. The optimizer was a memetic algorithm which combined a real-coded EA and the the variable neighbourhood search algorithm [604], which searches in increasingly larger local neighbourhoods around the current iterate.

In [811] the authors considered the problem of optimizing a robust aircraft control system using a memetic algorithms. The problem was formulated as a quadratic minimization problem where the goal was to find a set of matrix elements which optimize a prescribed system robustness measure. The memetic algorithm combined an EA with a hill-climbing local search.

In a multiobjective formulation [835] proposed a memetic algorithm for robust optimization while considering both the expected value and variance of the objective function. The study applied the algorithm to robust optimization of airfoils where the goal was to identify an airfoil shape yielding a low drag (aerodynamic friction) over a range of aircraft velocities. The proposed algorithms used a variant of the NSGA-II algorithm [196] to approximate the Pareto front and then invoked a gradient-based local search to refine solutions. For each solution the local search minimized one function at a time while treating the other as a constraint, and the resulting vector was used as a starting point for subsequent steps, repeating the procedure for the two objective functions.

In another multiobjective study [691] used the Design-for-Six-Sigma (DFSS) approach which considers both the mean and variance of the objective and proposed using a particle swarm optimizer (PSO) to obtain the Pareto front of the mean–variance objectives. A follow-up study [690] then extended the idea to a memetic algorithm combining an EA as a global search algorithm and then using a finite-differences quasi-Newton local search to further refine the solutions, an approach termed memetic algorithm for robust solution search. The local search was applied to a certain percentage of the population chosen at random but without considering the variance of the fitness, that is, a single objective refinement of the solutions. The authors also applied an ageing operator which adjusted the expected mean fitness based on the duration an individual has survived.

Considering multiobjective optimization and robustness [324] proposed a multiobjective EA for robust and constrained optimization. The algorithm uses a micro-GA (that is, having a very small population) as a form of a local search to obtain the worst-case performance of candidate solutions. It also uses a tabu-like approach which restricts and guides the EA and periodic re-evaluation of cached solutions to reduce uncertainty regarding their fitness.

In [521] the authors addressed the problem of robust optimization when no a-priori information is known about the uncertainties. Commonly, algorithms assume some a-priori statistical distribution for the unknown uncertainties (for example Gaussian) but this can be unfounded. The authors proposed the *inverse robust evolutionary design methodology* which combines an EA with a constrained local search (performed by an SQP solver). The idea is to replace the classical problem (termed *forward optimization*) with *inverse optimization* which locates a target solution satisfying some prescribed criteria:

$$\begin{aligned} &\min\ f(x) - T \\ &\text{s.t.}\quad x_l \leqslant x \leqslant x_u \end{aligned} \tag{14.17}$$

where T is the target output performance. The authors proposed a single objective variant which considers only the robustness function (the maximum uncertainty a design variable handles before violating the worst-case performance), bi-objective (robustness function and objective function) and tri-objective which also considers the opportunity fitness.

In [98] the authors proposed a memetic algorithm for robust airline scheduling where the goal was to obtain a fleet assignment which accounts for flight re-timing and aircraft rerouting. Using a multiobjective approach the study considered two objectives: schedule reliability and schedule flexibility. The proposed algorithm used a tailored representation (the adjacency representation often used in traveling salesman problems) and multi-memes (multiple local searchs) to improve effectiveness. The algorithm used three variants of local search, each considering the schedule reliability, schedule flexibility or both. The study also used a host of additional features such as archiving and biased sampling (to encourage exploration).

Also in scheduling problems, [837] proposed a memetic algorithm for the stochastic capacitated vehicle routing problem (CVRP). The baseline CVRP is that of determining the sequence in which a fleet of vehicles visits spatially distributed customers such that some cost measure (time, distance) is minimized. In the stochastic CVRP the customer demands and travel costs are no longer crisp which motivates a robust approach. The proposed algorithm samples the objective function around a set of solutions and selects (based on the problem formulation) either the expected (mean) response or the worst-case (max). The algorithm refines solutions using a local search combined with tabu search.

Following the worst case performance approach to robust optimization [684] proposed a memetic algorithm designed for expensive objective functions. The algorithm builds upon the earlier genetic algorithm with robust solution searching schemes (GARSS) [895] in which a random perturbation was added to a chromosome before evaluation. In its single evaluation variant each chromosome was perturbed once while in the multiple evaluations variant it was perturbed repeatedly and the final fitness was taken either as the mean or worst of the perturbed set. Empirical tests show that the multiple evaluations variant was more reliable than the single evaluation one but obviously required more function evaluations which makes the algorithm inapplicable to expensive problems. As such, the authors proposed an algorithm which combines a max-min optimization strategy with a TR model-assisted approach and a Baldwinian updating scheme. The algorithm starts with an initial sample (random or by design of experiments) and uses the baseline GARSS algorithm with the worst fitness of the perturbed set taken as the chromosome fitness. The GARSS is run for several generations while evaluating the true (expensive) function and all vectors and associated fitness are cached. Next, each individual in the population is refined with a TR local search where the goal of the latter is to find the worst case performance. To reduce function evaluations the local search used RBF models which were trained using cached vectors adjacent to the TR centre and the TR procedure follows that described in 14.2. The goal of the local search was to find the worst case performance for each population member by solving the max-min problem

$$\begin{aligned} &\min \; f(x+x_c) \\ &\text{s.t.} \;\; x \in \Omega \end{aligned} \tag{14.18}$$

where x is the vector of perturbations, x_c is the baseline candidate and Ω is the feasible range of perturbations. The search was performed using an SQP solver and the TR iterations terminated after a prescribed k expensive function evaluations. If the TR local-search found a lower objective value then it replaced the fitness of the original population members (that is, before the local search was invoked) in a Baldwinian learning scheme (the chromosome was not changed). Algorithm 26 gives a pseudocode of the framework (adapted from [684]). Performance analysis was based on a robust airfoil shape optimization problem with a parametrization

Algorithm 26. Trust-region Enabled Max-Min Surrogate-Assisted EA [684]

```
initialize database;
repeat
    for each individual i in population do
        if status is database building then
            evaluate individual with true (expensive) function and cache;
        endif
        else improve individual with TR–SQP search
            initialize TR;
            repeat
                train local RBF model using neighbours from database;
                establish domain where uncertain variables vary $\Omega$;
                find point of worst-fitness in TR using RBF models (trial step);
                evaluate predicted point with expensive function and cache it;
                update TR based on success of trial step;
            until TR termination condition met ;
            set individual's fitness to worst-case value;

    endfor
    Apply standard EA operators to create a new population;
until EA termination condition met ;
```

resulting in a 24-dimensional problem. The authors first obtained an airfoil shape without considering any perturbation (a classical non-robust optimization) as a reference shape. Next, they applied the framework to robust optimization in the presence of manufacturing errors ($\pm 5\%$ error bounds on design variables). Analysis showed the performance of the robust airfoil is indeed more stable than that of the non-robust one. The authors also optimized the airfoil for perturbations in operating conditions (cruise velocity). Similarly to the previous case, the robust airfoil performance was more stable over the entire range of cruise speeds while the performance of the non-robust airfoil degrades quickly outside the nominal operating point. Overall, the framework was able to generate robust designs on a limited computational budget.

14.4 Uncertainty Due to Noise

In many real-world applications repeatedly evaluating the same vector returns slightly different objective values, a scenario termed *noisy* optimization. Such fluctuations in the response imply uncertainty regarding the true function value. Noisy functions are encountered in two main scenarios:

1. The response is obtained by measuring some real-world process and noise is either inherent in the process or in the measurement instruments. For example, reading electrical signals from an electric motor [104].

2. The objective function depends on some random process. For example, when optimizing the topology of neural networks the same vectors (candidate topologies) can produce different responses due to random initialization of network weights [949].

The dominant (and sometimes implicit) assumptions in noisy optimization problems are that the noise is random (so it cannot be filtered out a-priori) and that its amplitude is much smaller than the underlying objective response (so it only moderately deforms the landscape). Many studies also assume that the noise is Gaussian.

Since the observed responses are corrupted by noise some additional sampling mechanism needs to be introduced to estimate the true objective value. These mechanisms come in two main flavours:

1. Explicit Averaging: a better estimate of the true response can be obtained by using multiple samples. In temporal sampling the same vector is re-sampled n times and under the assumption of random Gaussian noise this allows to improve (reduce) the estimated response variance by $\sqrt{n}$ [808]. A complementary approach is that of spatial sampling where the samples are taken from neighbouring points around the current individual [788].
2. Implicit Averaging: simply increasing the population size provides more samples of the objective function and implicitly combats noise. The population size can be either fixed (set a-priori to a high value) or adapted during the search.

With the first category (noise due to external processes), in [104] the authors tackled the problem of optimizing the control system of an electric motor. They used online optimization, that is, where each candidate control settings were tested in real-time and the resulting performance was fed back into the algorithm. The measurements of the motor were inherently noisy and to combat noise the study proposed a memetic algorithm which monitored the population diversity to control the degree of mutation: high diversity invoked more local searches while low diversity invoked a higher mutation rate. Also, the algorithm selected between two types of local search (Hooke-Jeeves pattern search [391] and Nelder-Mead simplex [653]) to refine vectors.

Also in this category, [462] proposed a memetic algorithm combining a real-coded EA with the Bacteria Foraging local search. The latter is inspired by the swim pattern of the E. coli bacteria in the presence of favourable/hostile environment (rich/poor with nutrients). The idea is to perform tentative moves (similar to the bacteria's swim pattern) and adapt the step size based on the success/failure of these moves. The authors applied the memetic algorithm to optimization of a Proportional/Integral/Derivative (PID) controller for an automatic voltage controller subject to a sine wave noise.

In [601] the authors proposed a memetic algorithm based on differential evolution where the scale factor was adjusted with a local search. The algorithm also employed a noise analysis component to determine whether to replace a parent with an offspring. Specifically, it compared the samples of fitness (for each) to determine whether the means were sufficiently distinct (so a comparison is meaningful). If so, the better solution was retained but otherwise the algorithm sampled more points and repeated the comparison.

In [43] the authors considered the noisy pattern recognition problem of inexact graph matching, that is, determining whether two images match when one is corrupted by noise. They proposed a memetic algorithm in a combinatorial framework where each graph is represented by a chromosome of its vertices. The GA uses tournament selection and a new position based cross-over but no mutation. A tailored local search explored the neighbourhood of a solution and if it succeeded in locating a better individual then the latter replaced the original in a Lamarckian update scheme. The operators were designed to be insensitive to vertex location to provide better immunity to noise.

Also in this class, [695] studied the problem of matching an input image to one from an available data set. The difficulty being that the input image may be partially obscured, deformed and so on which results in a noisy optimization problem. They used a specialized encoding to represent both the input image and the database images by segmenting them into lines and connecting angles. They proposed a memetic algorithm which combined a real-coded EA (one point cross-over, uniform mutation) and a hill-climbing local search. For each database image the algorithm matched each segment to the those of the input image while ignoring small differences (to combat minor image deformations).

Related to the second category of noise due to a random process, [171] proposed an EA which uses a self-organizing map (SOM) [477] as a local search operator. The algorithm was designed to solve the vehicle routing problem (VRP) with emphasis on noisy data. The SOM was used to allow immunity to noise and to fluctuations in customer demands. The authors have also proposed several dedicated operators which work in conjunction with the SOM to improve the search.

In [656, 660] the authors tackled the problem of training a neural network used for controlling resource discovery in peer-to-peer (P2P) networks. They considered a multi-layer perceptron (MLP) network with a topology of 22 input neurons and 10 hidden-layer neurons plus a bias channel resulting in 298 weights to optimize. Since the network needs to operate under a variety of query conditions this results in a noisy objective function. The authors proposed the adaptive global–local memetic algorithm (AGLMA) which combined a real-coded EA with self-adaptation and two local searches: the stochastic simulated-annealing (SA) [468] and the deterministic Hooke-Jeeves. To combat noise the algorithm adjusted the objective response by explicit averaging. The proposed algorithm used a measure of population diversity

$$\psi = 1 - \frac{\hat{F}_{\text{avg}} - \hat{F}_{\text{best}}}{\hat{F}_{\text{worst}} - \hat{F}_{\text{best}}} \tag{14.19}$$

where the measures are the average, best and worst fitness values in the population at the end of a generation. It follows that $\psi \simeq 1$ indicates high diversity and $\psi \simeq 0$ low diversity. The algorithm used this diversity measure to determine when to invoke each local search by the heuristic rules

$$\psi \begin{cases} \in [0.1, 0.5] & \text{invoke simmulated annealing} \\ < 0.2 & \text{invoke Hooke-Jeeves} \end{cases} \tag{14.20}$$

The idea is to use an explorative search (the SA) when the population diversity is decreasing (low ψ values). The Hooke-Jeeves local search was applied to the best individual and does not have the same explorative qualities but is more localized. It follows that for $\psi \in [0.1, 0.2]$ both local searchs are applied. The algorithm also leveraged on implicit resampling by adjusting the population size based on the diversity measure using the rule

$$S_{\text{pop}} = S^{f}_{\text{pop}} + S^{v}_{\text{pop}} \cdot (1 - \psi) \tag{14.21}$$

where S^{f}_{pop}, S^{v}_{pop} are a prescribed lower and upper bounds on the population size, respectively. Algorithm 27 gives a pseudo-code of the framework. As mentioned, the authors applied the algorithm to the topology optimization of a P2P network and benchmarked it against the Checkers Algorithm (CA), the Adaptive Checkers Algorithm (ACA) and a baseline real-coded GA while the optimization budget was $1.5e6$ function evaluations. The proposed algorithm (AGLMA) performed best, closely followed by ACA and lastly the CA and baseline GA. Although the AGLMA converged more slowly than the CA it obtained a better final solution. The paper explains that the ACA can be viewed as an AGLMA without the memetic (that is, local search) component which explains its slightly degraded performance and highlights the merit of the memetic approach. Further, the AGLMA and ACA also effectively filtered noise which was evident from the convergence analysis (given in the paper) when compared to the CA and baseline GA. Overall, the AGLMA framework was able to handle this high-dimensional and noisy optimization problem.

14.5 Uncertainty Due to Time-Dependency

In the three categories covered so far (expensive evaluations, robustness, noise) the underlying optimization problem was fixed. However, in many real-world applications the problem is time-dependant so (14.1) becomes

$$\begin{aligned} \min\ & f(x,t) \\ \text{s.t.}\ & g_i(x,t) \leqslant 0, \quad i = 1 \ldots k \\ & t = 1, 2, \ldots \end{aligned} \tag{14.22}$$

that is, the objective landscape, constraints and hence the problem optima may vary with time. Such problems arise in diverse applications such as scheduling [525] and control [755].

Algorithm 27. Adaptive Global–Local Memetic Algorithm [660]

```
1  sample weights $W$ and self-adaptive parameters $h$;
2  evaluate fitness of initial population with explicit averaging;
3  calculate merit value: $\psi \leftarrow 1 - \frac{\hat{F}_{\text{avg}} - \hat{F}_{\text{best}}}{\hat{F}_{\text{worst}} - \hat{F}_{\text{best}}}$;
4  while budget conditions and $\psi > 0.01$ do
5      for all individuals $i$ do
6          for all variables $j$ do
7              update weights and self-adaptive parameters;
8          endfor
9      endfor
10     evaluate fitness of population by explicit averaging;
11     sort population (parents+offspring) based on fitness;
12     if $0.1 \leqslant \psi \leqslant 0.5$ then
13         execute simmulated annealing on $2nd$ best individual;
14         if $\psi < 0.2$ then
15             execute Hooke-Jeeves on best individual;
16         endif
17         if simmulated annealing successful then
18             execute Hooke-Jeeves on individual improved by SA;
19         endif
20     endif
21     calculate $S_{\text{pop}} \leftarrow S^f_{\text{pop}} + S^v_{\text{pop}} \cdot (1 - \psi)$;
22     select $S_{\text{pop}}$ best individuals as the next generation;
23     calculate merit value: $\psi \leftarrow 1 - \frac{\hat{F}_{\text{avg}} - \hat{F}_{\text{best}}}{\hat{F}_{\text{worst}} - \hat{F}_{\text{best}}}$
24 endw
```

The time-dependant nature of such problems introduces several specific algorithmic considerations:

1. Since the optimization algorithm effectively needs to solve not one but a series of problems it should not drive the population of candidate solutions to fast convergence but should rather maintain diversity to allow the population to adapt to the changing landscapes.
2. Between subsequent time steps changes to the problem formulation are often small so it may be beneficial to search in the vicinity of the recent optimum (optimum tracking).

Due to their unique nature dynamic problems are often tested with a tailored suite of problems termed the Moving Peaks [82, 613, 906] which define a time-varying multimodal landscape where peaks deform and translate. There are also specific performance measures for dynamic problems where the commonly used one being the mean offline performance

$$f_{\text{off}} = \frac{1}{T} \sum_{t=1}^{T} f^*(t) \tag{14.23}$$

where $f^*(t)$ is the best objective value found at time step t [927].

In [904, 905, 906] the authors proposed a memetic algorithm combining a binary EA with the variable local search (VLS) operator to track optima in dynamic problems. The EA invoked the operator when the averaged best performance of the population dropped below a prescribed threshold. Once a change in the landscape was detected the VLS operator enabled a local search around individuals from the pre-change population, an approach motivated by the assumption that changes are gradual (as mentioned above). The extent of the search was variable and calibrated based on the observed degree of change. When the VLS operator was invoked the evolutionary operators of recombination and mutation were temporarily suspended and the EA generated new vectors by adding or subtracting (with equal probability) from the population a random binary vector (whose range of values was limited to define a small search neighbourhood). After a single application of the VLS the EA reverted back to standard recombination and mutation and observed the performance of the population elites over a period of several generations. If the mean performance did not reach its previous (pre-change) value then the range of the VLS operator was increased and the process was repeated.

In [915] the authors proposed a memetic algorithm based on a particle swarm optimizer (PSO) and a hill-climbing local search. The algorithm combined several techniques to improve its performance in dynamic problems:

1. when updating a particle's position the algorithm considered the best solution found by the particle and its neighbours (termed local-PSO) to avoid rapid convergence
2. particles were refined by a local search which stochastically perturbed an elite vector to perform a neighbourhood search
3. particles were positioned on a virtual 'ring' and communicate only with their ring-wise neighbours (irrespective of the Euclidean distance in the search space) as an additional measure to avoid rapid convergence and lastly
4. to increase diversity the worst solutions were extracted and allowed to evolve in a sub-swarm independently from the main swarm.

In [635] the authors proposed a memetic algorithm which combined the Extremal Optimization algorithm (EO) [77] and a deterministic local search. The former (EO) starts from a baseline solution and perturbs it to generate a population and then probabilistically eliminates the worse member. As such, it aims not for fast convergence but for gradual adaption, which has motivated the authors to apply it to dynamic problems. In a follow-up study [633] the authors proposed another variant which at each generation refined one population member with a local search using the Hooke-Jeeves algorithm. Another follow-up study [634] evaluated both the EO with the Hooke-Jeeves variant and with an improved local search which scanned along each coordinate with an initial step and adjusted the step size depending on the search progress.

In [232] the authors tackled dynamic and highly constrained problems and proposed a memetic algorithm based on the scatter search framework [321] which combines a global search (diversification) and a local search (intensification). The global search generated solutions similarly to an evolutionary recombination operator and where an offspring could replace only a parent. Solutions were also generated in a Nelder-Mead simplex-like move which explored along promising directions. Next, solutions were chosen for refinement based on competitive ranking (considering both their fitness and diversity) and were refined with one of several local optimizers (the authors considered variants of SQP and hill-climbing). The algorithm handled constraints by a static penalty method.

Recently [482] proposed a memetic algorithm for dynamic multiobjective problems. The idea is to accelerate the convergence of a multiobjective EA (or similar algorithms) by predicting the change in the Pareto set based on the observed pattern in past time steps under the assumption that the Pareto set does not change erratically but follows an identifiable pattern. The approach used a *predictive gradient* (g) which approximated the shift in the population between consecutive time-steps. The idea was then to shift individuals in the population using the rule

$$x_{\text{new}} = x + \mu g \,. \tag{14.24}$$

The predictive gradient was calculated based on changes in the centroid of the non-dominated solutions. The algorithm monitored landscape changes by comparing the fitness of a subset of individuals and so a mismatch between consecutive time-steps indicated a landscape change. This then triggered a population update where a pre-determined number of individuals were randomly selected and updated with the predictive gradient. The approach was implemented within a multiobjective evolutionary gradient search algorithm.

In [913] the authors proposed a memetic algorithm for dynamic optimization which used a binary representation where at each generation the elite was refined by a local search and added several tailored enhancements. First, it used two hill-climbing variants for the local search:

1. greedy crossover hill climbing (GCHC): used the current elite and another parent (chosen by roulette wheel selection) and generated an offspring by uniform crossover and
2. steepest mutation hill climbing (SMHC): the elite individual was mutated by randomly flipping its bits.

In both variants the offspring replaced the elite if it was better. Another feature was that the algorithm adapted the probability of applying each variant based on their success in previous steps (starting from an equal probability of 0.5 for both). The success of a step was measured by

$$\eta = \frac{|f_{\text{imp}} - f_{\text{ini}}|}{f_{\text{ini}}} \tag{14.25}$$

where f_{imp} , f_{ini} were the improved and initial objective values, respectively, and the probability of applying each variant was updated by

$$p(t+1) = p(t) + \Delta \cdot \eta(t) \tag{14.26}$$

where Δ was prescribed by the user. Lastly, the algorithm safeguarded the population diversity using two procedures:

1. adaptive dual mapping (ADM): before starting a local search the method evaluated the bit-complementary of the initial solution and used the better of the two as the resultant initial vector and
2. triggered random immigrants (TRI): when the population diversity was deemed low a portion of the population was replaced by randomly generated individuals while the population diversity was measured by

$$\xi = \frac{\sum_{i=1}^{s} d(x^\star, x_i)}{s} \tag{14.27}$$

   where s is the population size and $d(x^\star, x_i)$ is the Euclidean distance between the current elite and the ith individual in the population.

Algorithm 28 gives a pseudo-code of the framework. The authors evaluated the proposed framework using tests derived from stationary problems (the 100-bit binary coded variants of the OneMax, Plateau, RoyalRoad and Deceptive). The authors used memetic variants with the GCHC, SMHC, AHC operators described above, a

Algorithm 28. Memetic Algorithm for Dynamic Problems [913]

```
initialize population and evaluate individuals;
calculate algorithm parameters;
select elite for local search;
if ADM is used then
    create a dual of the elite and evaluate;
    if dual is better set as new elite;
endif
perform AHC with elite;
repeat
    apply standard EA operators(selection,recombination,mutation) to create offspring;
    evaluate offspring and select individuals for next generation;
    select elite for local search;
    calculate algorithm parameters;
    if ADM is used then
        create a dual of the elite and evaluate;
        if dual is better set as new elite;
    endif
    perform AHC with elite;
    if TRI is used then
        if $\xi < \theta_0$ then
            replace a prescribed number of worst individuals in new generation with
            random immigrants;
        endif
    endif
until stop condition is met ;
```

baseline GA, a baseline GA with population restart when a change is detected, a GA with random immigrants, a GA with elitism-based immigrants and the population-based incremental algorithm (PBIL). Performance analysis indicated that:

1. the diversity-based procedures improved performance in dynamic problems
2. the ADM approach performed better when there were significant changes in the environment while the TRI performs better in correnspondence to small changes
3. the optimal local search was problem dependant and there was no clear winner and lastly
4. the AHC approach used multiple local searches which provided more robustness.

Overall, results indicated that the combination of the AHC as a local search with ADM and TRI provided an effective memetic framework for dynamic problems.

14.6 Conclusion

Optimization problems arising in real-world applications can differ significantly from synthetic mathematical test problems and one such major difference is uncertainty induced by approximation, robustness, noise or time-dependency. While computational intelligence algorithms have been applied to such problems, memetic algorithms offer enhanced capabilities which significantly improve search efficacy under such challenging settings, as surveyed in this chapter. The complexity of real-world problems can be expected to grow, for example, to problems with multiple uncertainties (expensive and robust or noisy and dynamic). In such settings memetic algorithms will likely further establish their standing as a potent framework for optimization in the presence of uncertainties.

Acknowledgements. This research is kindly supported by the Japan Society for Promotion of Science.