Algorithmica
https://doi.org/10.1007/s00453-018-0462-1

On the Analysis of Trajectory-Based Search Algorithms: When is it Beneficial to Reject Improvements?

Samadhi Nallaperuma[1] · **Pietro S. Oliveto**[1] · **Jorge Pérez Heredia**[1,2] · **Dirk Sudholt**[1]

Received: 30 June 2017 / Accepted: 29 May 2018

Abstract We investigate popular trajectory-based algorithms inspired by biology and physics to answer a question of general significance: when is it beneficial to reject improvements? A distinguishing factor of SSWM (strong selection weak mutation), a popular model from population genetics, compared to the Metropolis algorithm (MA), is that the former can reject improvements, while the latter always accepts them. We investigate when one strategy outperforms the other. Since we prove that both algorithms converge to the same stationary distribution, we concentrate on identifying a class of functions inducing large mixing times, where the algorithms will outperform each other over a long period of time. The outcome of the analysis is the definition of a function where SSWM is efficient, while Metropolis requires at least exponential time. The identified function favours algorithms that prefer high quality improvements over smaller ones, revealing similarities in the optimisation strategies of SSWM and Metropolis respectively with best-improvement (BILS) and first-improvement (FILS) local search. We conclude the paper with a comparison of the performance of these algorithms and a (1,λ) RLS on the identified function. The algorithm favours the steepest gradient with a probability that increases with the size of its offspring population. The results confirm that BILS excels and that the (1,λ) RLS is efficient only for large enough population sizes.

Keywords Theory · Evolutionary algorithms · Non-elitism · Metropolis algorithm · Strong selection weak mutation regime

✉ Dirk Sudholt
d.sudholt@sheffield.ac.uk

1 University of Sheffield, Sheffield S1 4DP, UK

2 Basque Center for Applied Mathematics, 48009 Bilbao, Spain

Published online: 11 June 2018

1 Introduction

The Strong Selection Weak Mutation (SSWM) algorithm is a recent randomised search heuristic inspired by the popular model of biological evolution in the ‘strong selection, weak mutation regime’ [14,15]. The regime applies when mutations are rare and selection is strong enough such that new genotypes either replace the parent population or are lost completely before further mutations occur [5,7].

The SSWM algorithm belongs to the class of trajectory-based search heuristics that evolve a single trajectory of search points rather than using a population. Amongst single trajectory algorithms, well-known ones are (randomised) local search, simulated annealing, the Metropolis algorithm (MA)—simulated annealing with fixed temperature—and simple classes of evolutionary algorithms such as the well-studied $(1+1)$ EA and the $(1+\lambda)$ EA. The main differences between SSWM and the $(1+1)$ EA is that the latter only accepts new solutions if they are at least as good as the previous ones (a property called *elitism*), while SSWM can reject improvements and it may also accept non-improving solutions with some probability (known as *non-elitism*). This characteristic may allow SSWM to escape local optima by gradually descending the slope leading to the optimum rather than relying on large, but rare, mutations to a point of high fitness far away.

A recent study has rigorously analysed the performance of SSWM in comparison with the $(1+1)$ EA for escaping local optima [11]. The study only allowed SSWM to use local mutations such that the algorithm had to rely exclusively on its non-elitism to escape local optima, hence to highlight the differences between elitist and non-elitist strategies. A vast class of fitness functions, called fitness valleys, was considered. These valleys consist of paths between consecutive local optima where the mutation probability of going forward on the path is the same as going backwards. However, the valleys may have arbitrary length and arbitrary depth, where the length is measured by the hamming distance while the depth is the maximal fitness difference that has to be overcome.

The analysis revealed that the expected time of the $(1+1)$ EA to cross the valley (i.e. escape the local optimum) is exponential in the length of the valley while the expected time for SSWM can be exponential in the depth of the valley.

However, other non-elitist trajectory-based algorithms such as the well-known Metropolis algorithm have the same asymptotic runtime as SSWM on fitness valleys, independent of lengths and depths. While both algorithms rely on non-elitism to descend the valleys, it is not necessarily obvious that the algorithms should have the same runtime on the valleys, because they differ significantly in the probability of accepting improving solutions. In particular, Metropolis always accepts improvements while SSWM may reject an improving solution with a probability that depends on the difference between the quality of the new and the previous solution.

In this paper we investigate SSWM and Metropolis with the goal of identifying function characteristics for which the two algorithms perform differently. Given that the main difference between the two is that SSWM may reject improvements, we aim to identify a class of functions where it is beneficial to do so and, as a result, identify an example where SSWM outperforms Metropolis.

The roadmap is as follows. After introducing the algorithms precisely in the Preliminaries section, we show in Sect. 3 that our task is not trivial by proving that both algorithms converge to the same stationary distribution for equivalent parameters. While this result seems to have been known in evolutionary biology [17] we are not aware of a previous proof in the literature. In Sect. 4 we define a simple fitness function (called *3 state model*) where two possible choices may be made from the initial point; one leading to a much larger fitness than the other. The idea is that, while Metropolis should be indifferent to the choice, SSWM should pick one choice more often than the other. Although this intuition is true, it turns out that, due to Metropolis' ability of escaping local optima, the mixing time for the 3 state model is small and afterwards the two algorithms behave equivalently as proven in the previous section. In Sect. 5 we extend the fitness function (leading to a *5 state model*) by adding two more states of extremely high fitness such that, once the algorithms have made their choice, the probability of escaping the local optima is very low. By tuning these high fitness points we can either reward or penalise a strategy that rejects small improvements. We capitalise on this by concatenating several 5 state models together (each of which we refer to as a *component*) and by defining a composite function that requires that a high number of correct choices are made by the algorithm. Then we show that for appropriate fitness values of the different states, SSWM achieves the target of the function and Metropolis does not with overwhelming probability. We complement our theoretical findings with experiments which help to understand the complete picture.

In Sect. 6 we consider other common single trajectory based search algorithms to compare their performance on the identified function class with SSWM and Metropolis. The reason that SSWM outperforms Metropolis for the identified composite function is that the former algorithm tends to favour the acceptance of search points on the slope of largest uphill gradient while the latter algorithm accepts any improvement independent of its quality. Hence, we expect that also other algorithms that prefer improvements of higher quality over smaller ones (i.e., a characteristic often referred to as *exploitation*) perform well on the composite function. To this end we consider the well known Best-Improvement Local Search (BILS) algorithm that always selects the neighbouring point of highest fitness and compare it with a less exploitational local search strategy which accepts the first found improvement (FILS). Finally, we also consider a classical single trajectory evolutionary algorithm that favours exploitation. In order to achieve a fair performance comparison with SSWM and Metropolis we consider the (1,λ) RLS algorithm which, like the former algorithms, uses non-elitism and local mutations. The results show that BILS excels on the composite function while the (1,λ) RLS only works for large enough population sizes.

This article extends a previous conference paper [10] that only focussed on the comparison of SSWM and the Metropolis algorithm.

2 Preliminaries

As mentioned in the introduction, we will be considering trajectory-based heuristics. The pseudo-code of Algorithm 1 considers algorithms with local mutations, i.e., only search points that differ in one bit can be sampled. However, the new individual will be accepted or rejected according to a probability function known as the acceptance probability $p_{\mathrm{acc}} : \mathbb{R} \to [0, 1]$.

Algorithm 1 General trajectory-based Algorithm

```
Initialise $x \in \{0, 1\}^n$
repeat
  $y \leftarrow$ flip uniformly at random one bit from $x$
  $\Delta f = f(y) - f(x)$
  Choose $r \in [0, 1]$ uniformly at random
  if $r \leq p_{\mathrm{acc}}(\Delta f)$ then
    $x \leftarrow y$
  end if
until stop
```

Two important characteristics of the acceptance probability are how detrimental and beneficial moves are dealt with. Elitist algorithms such as RLS will directly reject any worsening move and accept any improving search point. Hence, an elitist trajectory-based algorithm will not be able to escape local optima.

To avoid this weakness, the algorithm must relax its selection strength. This is the case in the Metropolis [9] algorithm where detrimental moves are allowed with some probability, depending on the temperature $1/\alpha$. However, improvements will always be accepted regardless of their magnitude:

$$p_{\mathrm{acc}}^{\mathrm{MA}}(\Delta f) = \begin{cases} 1 & \text{if } \Delta f \geq 0 \\ e^{\alpha \Delta f} & \text{if } \Delta f < 0 \end{cases} \tag{1}$$

To investigate the other main characteristic of non-elitism, allowing the rejection of improvements, we will study a recently introduced algorithm [11,15,16] based on the so called SSWM evolutionary regime from Population Genetics (PG). Within this regime a new genotype will eventually take over of a population of size $N \in \mathbb{N}^+$ or become extinct according to the following expression. This formula depends on the fitness difference Δf and a scaling factor $\beta \in \mathbb{R}^+$ [7]. To cast this regime as an algorithm we simply use the following acceptance probability in Algorithm 1. For $\Delta f \neq 0$ we define

$$p_{\mathrm{acc}}^{\mathrm{SSWM}}(\Delta f) = p_{\mathrm{fix}}(\Delta f) = \frac{1 - e^{-2\beta \Delta f}}{1 - e^{-2N\beta \Delta f}} \tag{2}$$

and $p_{\mathrm{acc}}^{\mathrm{SSWM}}(0) := \lim_{\Delta f \to 0} p_{\mathrm{acc}}^{\mathrm{SSWM}}(\Delta f) = 1/N$. Figure 1 presents an example of these two acceptance probabilities. We observe how both algorithms treat worsening moves similarly. The main difference arises when dealing with improvements. Unlike Metropolis, SSWM will prefer to keep the current search point against a small improvement (until values of Δf that make $p_{\mathrm{fix}} \geq 1/2$). However when the fitness difference is large enough the algorithm will be satisfied to move to the new solution. This is the crucial feature that we will be exploiting in the following sections.

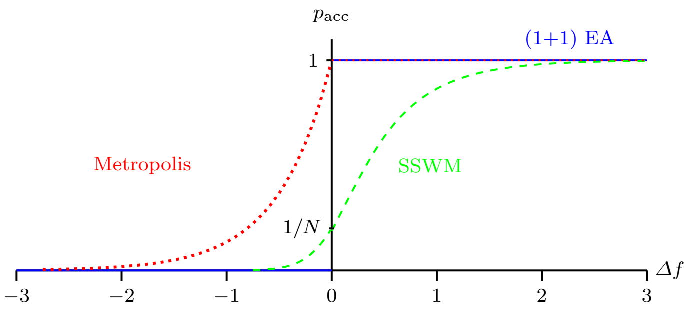


Fig. 1 Acceptance probability for the $(1+1)$ EA (blue solid line), Metropolis (red dotted line) and SSWM (green dashed line) (Color figure online)

3 A Common Stationary Distribution

We first show that SSWM and Metropolis have the same stationary distribution, starting by briefly recapping the foundations of Markov chain theory and mixing times (see, e. g. [1,6,8]). A Markov chain is called *irreducible* if every state can be reached from every other state. It is called *periodic* if certain states can only be visited at certain times; otherwise the chain is *aperiodic*. Markov chains that are both irreducible and aperiodic are called *ergodic* and they converge to a unique stationary distribution π.

Theorem 1 *Consider SSWM and Metropolis with local mutations over a Markov chain with states $x \in \{0,1\}^n$ and a fitness function $f : \{0,1\}^n \to \mathbb{R}$. Then the stationary distribution of such process will be*

$$\pi(x) = \frac{e^{\gamma f(x)}}{Z}$$

where $Z = \sum_{x\in\{0,1\}^n} e^{\gamma f(x)}$ and $\gamma = 2(N-1)\beta$ in the case of SSWM and $\gamma = \alpha$ for Metropolis.

Proof First note that the acceptance probability of Metropolis has the following property: $p_{\mathrm{acc}}(\Delta f)/p_{\mathrm{acc}}(-\Delta f) = e^{\gamma \Delta f}$. This relation is also true for SSWM with $\gamma = 2\beta(N-1)$ (Lemma 2 in [15]). The stationary condition for a distribution $\pi(x)$ can be written as (cf. Proposition 1.19 in [8])

$$\pi(x)\cdot p(x\to y) = \pi(y)\cdot p(y\to x), \quad \text{for all } \ x, y \in \{0,1\}^n$$

where $p(x\to y)$ is the probability of moving to state y given that the current state is x. Therefore

$$\pi(x)\cdot p(x\to y)$$
$$= \frac{e^{\gamma f(x)}}{Z}\cdot\frac{1}{n}\cdot p_{\mathrm{acc}}(f(y)-f(x))$$

$$= \frac{e^{\gamma f(x)}}{Z} \cdot \frac{1}{n} \cdot \frac{p_{\text{acc}}(f(y) - f(x))}{p_{\text{acc}}(f(x) - f(y))} \cdot p_{\text{acc}}(f(x) - f(y)),$$

since $p_{\text{acc}}(\Delta f)/p_{\text{acc}}(-\Delta f) = e^{\gamma \Delta f}$ we obtain

$$\begin{aligned} \pi(x) \cdot p(x \to y) &= \frac{e^{\gamma f(x)}}{Z} \cdot \frac{1}{n} \cdot e^{\gamma(f(y)-f(x))} \cdot p_{\text{acc}}(f(x) - f(y)) \\ &= \frac{e^{\gamma f(y)}}{Z} \cdot \frac{1}{n} \cdot p_{\text{acc}}(f(x) - f(y)) \\ &= \pi(y) \cdot p(y \to x). \end{aligned}$$

□

The distance between the current distribution and the stationary distribution is measured as follows by the *total variation distance*. For two distributions μ and ν on a state space Ω it is defined as

$$||\mu - \nu|| = \frac{1}{2} \sum_{x \in \Omega} |\mu(x) - \nu(x)| = \max_{A \subseteq \Omega} |\mu(A) - \nu(A)|$$

where the last equality is well known (see, e. g. Proposition 4.2 in [8]). Now the mixing time is defined as the first point in time where the total variation distance decreases below $1/(2e)$ (the constant $1/(2e)$ being a somewhat arbitrary choice in [20]).

Definition 1 (*Mixing time* [20]) Consider an ergodic Markov chain starting in x with stationary distribution π. Let $p_x^{(t)}$ denote the distribution of the Markov chain after t steps. Let $t_x(\varepsilon)$ be the time until the total variation distance between the current distribution and the stationary distribution has decreased to ε: $t_x(\varepsilon) = \min\{t : ||p_x^{(t)} - \pi|| \le \varepsilon\}$. Let $t(\varepsilon) := \max_{x \in \Omega} t_x(\varepsilon)$ be the worst-case time until this happens.

The mixing time t_{mix} of the Markov chain is then defined as $t_{\text{mix}} := t(1/(2e))$.

After the mixing time, both algorithms will be close to the stationary distribution, hence any differing behaviour can only be shown before the mixing time. In the following, we aim to construct problems where the mixing time is large, such that SSWM and Metropolis show different performance over a long period of time. In particular, we seek to identify a problem where the expected first hitting time of SSWM is less than the mixing time.

4 A 3 State Model

We first introduce a fitness function defined on 2 bits. We will analyse the behaviour of SSWM and Metropolis on this function, before proceeding (in Sect. 5.1) to concatenate n copies of the fitness function to create a new function where SSWM drastically outperforms Metropolis.

The idea is simple: we start in a search point of low fitness, and are faced with two improving moves, one with a higher fitness than the other. This construction requires

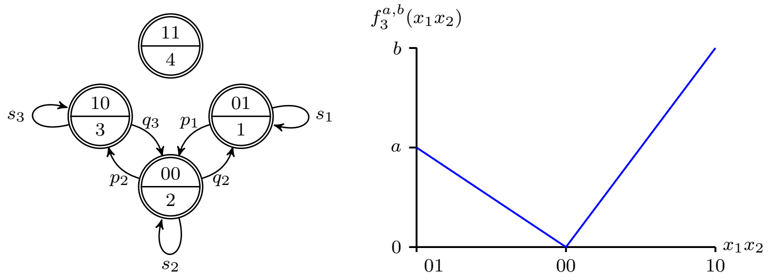


Fig. 2 Diagrams of the relevant nodes of $f_3^{a,b}(x_1x_2)$ at the genotype and phenotype level

3 search points, which are embedded in a 2-dimensional hypercube as shown in Fig. 2. The 4th possible bitstring will have a fitness of $-\infty$, making it inaccessible for both Metropolis and SSWM. As common in evolutionary computation, we sometimes refer to the model states as *phenotypes* and their bitstring encoding as *genotypes*.

Considering the 3 relevant nodes of the Markov Chain, they form a valley structure tunable through two parameters a and b representing the fitness difference between the minimum and the local and global optimum respectively.

Definition 2 (*3 state model*) For any $b > a > 0$ and a bit-pair $\{0, 1\}^2$ the 3 state model $f_3^{a,b}$ assigns fitness as follows:

$$\begin{aligned} f_3^{a,b}(01) &= a, && (\textit{state 1}) \\ f_3^{a,b}(00) &= 0, && (\textit{state 2}) \\ f_3^{a,b}(10) &= b, && (\textit{state 3}) \end{aligned}$$

and $f_3^{a,b}(11) = -\infty$.

This model is loosely inspired by a two-locus (two bit) Dobzhansky–Muller incompatibility model [13,21] in population genetics, where starting from an initial genotype (00 with fitness 0) there are two beneficial mutations (genotypes 01 with fitness $a > 0$ and 10 with fitness $b > 0$), but both mutations together are incompatible (genotype 11 with fitness $-\infty$).

This model is well suited for our purposes as Metropolis is indifferent to the choice of the local optimum (fitness $a > 0$) and the global optimum (fitness $b > a$), hence it will make either choice from state 00 with probability $1/2$. SSWM, on the other hand, when parameterised accordingly, may reject a small improvement of fitness a more often than it would reject a larger improvement of $b > a$. Hence we expect SSWM to reach the global optimum with a probability larger than $1/2$ in just a relevant step (an iteration excluding self-loops). We make this rigorous in the following.

Since the analysis has similarities with the classical Gambler's Ruin problem (see e.g. [3]) we introduce similar concepts to the ruin probability and the expected duration of the game.

Definition 3 (*Notation*) Consider a Markov Chain with only local probabilities

$$P(X_{t+1} = j \mid X_t = i) = \begin{cases} q_i & \text{if } j = i - 1 \\ s_i = 1 - q_i - p_i & \text{if } j = i \\ p_i & \text{if } j = i + 1 \\ 0 & \text{if } j \notin \{i - 1, i, i + 1\}. \end{cases}$$

Then, we define absorbing probabilities ρ_i as the probabilities of hitting state k before state 1 starting from i. Equivalently, we define expected absorbing times $\mathrm{E}\,(T_{k \vee 1} \mid i)$ as the expected hitting times for either state 1 or k starting from i.

Note that this definition may differ from the standard use of *absorbing* within Markovian processes. In our case the state k has an absorbing probability, but the state itself is not absorbing since the process may keep moving to other states.

The following lemma derives a closed form for the just defined absorbing probability, both for the general scheme, Algorithm 1, and for two specific algorithms. The obtained expression of $\rho_2 = p_2/(p_2 + q_2)$ is simply the conditional probability of moving to the global optimum p_2 given that the process has moved, hence the factor $p_2 + q_2 = 1 - s_2$ in the denominator.

Theorem 2 *Consider any trajectory-based algorithm that fits in Algorithm 1 on $f_3^{a,b}$ starting from state 2. Then the absorbing probability of state 3 is*

$$\rho_2 = \frac{p_2}{p_2 + q_2}.$$

And for Metropolis and SSWM ($N \geq 2$) it is

$$\rho_2^{\mathrm{MA}} = \frac{1}{2} \qquad \rho_2^{\mathrm{SSWM}} = \frac{p_{\mathrm{fix}}(b)}{p_{\mathrm{fix}}(b) + p_{\mathrm{fix}}(a)} > \frac{1}{2}.$$

Proof Let us start expressing the absorbing probability with a recurrence relation: $\rho_2 = p_2\rho_3 + q_2\rho_1 + (1 - p_2 - q_2)\rho_2$. Using the boundary conditions $\rho_3 = 1$ and $\rho_1 = 0$ we can solve the previous equation yielding $\rho_2 = p_2/(p_2 + q_2)$.

The result for Metropolis follows from introducing $p_2 = q_2$ since both probabilities lead to a fitness improvement. For SSWM the mutational component of p_2 and q_2 cancels out, yielding only the acceptance probabilities. Finally the lower bound of $1/2$ is due to state 3 having a fitness $b > a$. □

Note that SSWM's ability to reject improvements resembles a strategy of *best improvement* or *steepest ascent* [18]: since the probability of accepting a large improvement is larger than the probability of accepting a small improvement, SSWM tends to favour the largest uphill gradient. Metropolis, on the other hand, follows the first slope it finds, resembling a *first ascent* strategy.

However, despite these different behaviours, we know from Theorem 1 that both algorithms will eventually reach the same state. This seems surprising in the light of

Theorem 2 where the probabilities of reaching the local versus global optimum from the minimum are potentially very different.

This seeming contradiction can be explained by the fact that Metropolis is able to undo bad decisions by leaving the local optimum and going back to the starting point. Furthermore, leaving the local optimum has a much higher probability than leaving the global optimum. In the light of the previous discussion, Metropolis' strategy in local optima resembles that of a *shallowest descent*: it tends to favour the smallest downhill gradient. This allows Metropolis to also converge to the stationary distribution by leaving locally optimal states.

We show that the mixing time is asymptotically equal to the probability of accepting a move leaving the local optimum, state 1. Note that asymptotic notation is used with respect to said probability, as the problem size is fixed to 2 bits. To be able to bound the mixing time using Theorem 1.1 in [2], we consider *lazy* versions of SSWM and Metropolis: algorithms that with probability $1/2$ execute a step of SSWM or MA, respectively, and otherwise produce an idle step. This behaviour can also be achieved for the original algorithms by appending two irrelevant bits to the encoding of $f_3^{a,b}$.

Another assumption is that the algorithm parameters are chosen such that $\pi(3) \geq 1/2$. This is a natural assumption as state 3 has the highest fitness, and it is only violated in case the temperature is extremely high.

Theorem 3 *The mixing time of lazy SSWM and lazy Metropolis on $f_3^{a,b}$ is $\Theta(1/p_{\text{acc}}(-a))$, provided $b > a > 0$ are chosen such that $\pi(3) \geq 1/2$.*

Proof We use the transition probabilities from Fig. 2. According to Theorem 1.1 in [2], if $\pi(3) \geq 1/2$ then the mixing time of the lazy algorithms is of order $\Theta(t)$ where

$$t = \frac{1}{p_1} + \frac{\pi(1) + \pi(2)}{\pi(2) p_2}$$

As $p_1 = 1/2 \cdot p_{\text{acc}}(-a)$ this proves a lower bound $\Omega(1/p_{\text{acc}}(-a))$. For the upper bound, we bound t from above as follows, using $\pi(1)p_1 = \pi(2)q_2$ (the stationary distribution is reversible):

$$\begin{aligned} t &= \frac{1}{p_1} + \frac{\pi(1) + \pi(2)}{\pi(2) p_2} \\ &= \frac{1}{p_1} + \frac{\pi(1)}{\pi(2) p_2} + \frac{1}{p_2} \\ &= \frac{1}{p_1} + \frac{q_2}{p_2} \cdot \frac{1}{p_1} + \frac{1}{p_2} \\ &\leq \frac{3}{p_1} \end{aligned}$$

as $q_2/p_2 = p_{\text{acc}}(a)/p_{\text{acc}}(b) \leq 1$ and $p_2 \geq p_1$. Recalling that $p_1 = 1/2 \cdot p_{\text{acc}}(-a)$ completes the proof. □

4.1 Experiments

We performed experiments to see the analysed dynamics more clearly. To this end, we considered a concatenated function

$$f(X) = \sum_{i=1}^{n} f_3^{a,b}(x_i)$$

consisting of n copies of the 3 state model (i.e. n components) x_i with $1 \leq i \leq n$, such that the concatenated function $f(X)$ returns the sum of the fitnesses of the individual components. Note that $2n$ bits are used in total. In our experiments, we chose $n = 100$ components.

In the case of SSWM we considered different population sizes $N = (10, 100)$ and scaling parameter values $\beta = (0.01, 0.1)$. For Metropolis we choose a temperature of $1/\alpha$, such that $\alpha = 2(N-1)\beta$. This choice was made according to Theorem 1 such that both algorithms have the same stationary distribution. The algorithms are run for 10,000 iterations. The fitness values for states representing local and global optimum are chosen as $a = 1$ and $b = 10$ respectively. We record the average and standard deviations of the number of components in the local and global optimum for 50 runs.

Figure 3 shows the number of components optimised (at both state 1 or state 3) for SSWM and MA. As suggested by Lemma 2, we observe on the left graph how SSWM (green curve) outperforms MA which only optimises correctly half of the components (purple curve). However, we know from Theorem 1 that both algorithms will eventually reach the same state. This is shown on the right plot of Fig. 3 where the temperature was increased to facilitate the acceptance of worsening moves by MA.

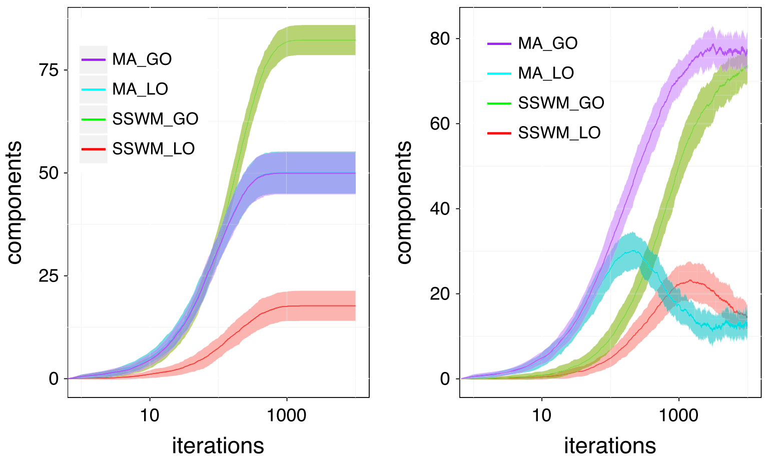


Fig. 3 Performance of SSWM with $N = 100$ and $\beta = 0.1$ (left) and $N = 10$ and $\beta = 0.01$ (right) on 100 concatenated components of the 3 state model. For Metropolis the temperature was chosen such that $\alpha = 2(N-1)\beta$ in both cases. The average number of components ($\pm$ one standard deviation) in the global and local optimum are plotted for SSWM and for Metropolis with colours red, green, purple and cyan respectively (Color figure online)

The reason why the limit behaviour is only achieved on the right hand plot of Fig. 3 is that the mixing time is inversely proportional to $p_{\text{acc}}(-a)$ (Theorem 3), which in turn depends on a and the parameters of SSWM and MA. If the temperature is low (large α), the algorithms show a different behaviour before the mixing time, whereas if the temperature is high (small α), the algorithms quickly reach the same stationary distribution within the time budget given.

5 A 5 State Model

We saw in the previous section how two algorithms with different selection operators displayed the same limit behaviour. Moreover the mixing time was small for both algorithms despite the asymmetric valley structure of the function. This asymmetry favoured moving towards the steepest slope, a landscape feature from which SSWM benefits and Metropolis is indifferent. However this feature also implied that it was easier climbing down from the shallowest slope, and Metropolis successfully exploits this feature to recover from wrong decisions.

Making use of these results we build a new function where the previous local optimum will now be a transition point between the valley and the new local optimum. We will assign an extremely large fitness to this new search point. In this this way we *lock in* bad decisions made by any of the two algorithms. In the same way, if the algorithm moves to the previous global optimum we offer a new search point with the highest fitness.

This new *5 state model* is shown in Fig. 4, along with its encoding of genotypes in a 3-dimensional hypercube.

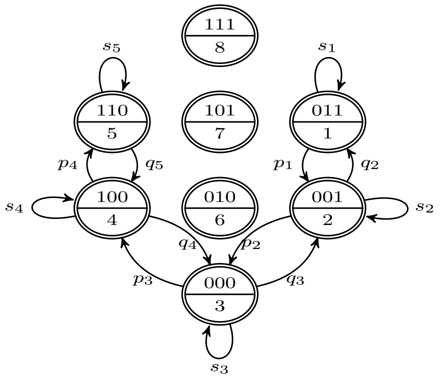
Definition 4 (*5 state model*) For any $M' > M \gg b > a > 0$, with $M' - b > M - a$ and a search point $x \in \{0, 1\}^3$ the 5 state model $f_5^{M,a,b,M'}$ assigns fitness as follows

$$\begin{aligned}
f_5^{M,a,b,M'}(011) &= M, && (\textit{state 1})\\
f_5^{M,a,b,M'}(001) &= a, && (\textit{state 2})\\
f_5^{M,a,b,M'}(000) &= 0, && (\textit{state 3})\\
f_5^{M,a,b,M'}(100) &= b, && (\textit{state 4})\\
f_5^{M,a,b,M'}(110) &= M' && (\textit{state 5})
\end{aligned}$$

and $f_5^{M,a,b,M'}(010) = f_5^{M,a,b,M'}(101) = f_5^{M,a,b,M'}(111) = -\infty$.

Let us consider the Markov chain with respect to the above model. For simplicity we refer to states with the numbers 1–5 as in the above description.

Again, we will compute the absorbing probability for the global optimum (state 5 or 110 of the Markov Chain). Note that by choosing very large values of M and M', we can make the mixing time arbitrarily large, as then the expected time to leave state 1 or state 5 becomes very large, and so does the mixing time.



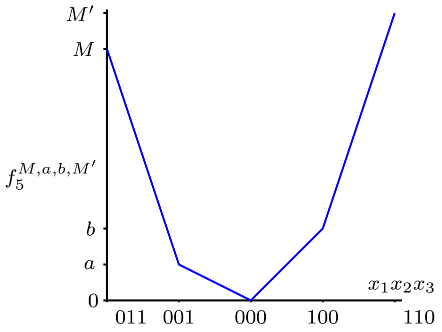


Fig. 4 Diagrams of the relevant nodes of $f_5^{M,a,b,M'}$ at the genotype and phenotype level

For simplicity we introduce the following conditional transition probabilities Q_i and P_i for each state i as

$$P_i := \frac{p_i}{p_i + q_i} \qquad Q_i := \frac{q_i}{p_i + q_i}. \tag{3}$$

By using this notation the following lemma derives a neat expression for the absorption probability $\rho_3 = P_3 P_4/(Q_2 Q_3 + P_3 P_4)$. This formula can be understood in terms of events that can occur in 2 iterations starting from state 3. Since Q and P are conditioning on the absence of self-loops there will be only 4 events after 2 iterations, whose probabilities will be $\{Q_3 Q_2, Q_3 P_2, P_3 Q_4, P_3 P_4\}$. Therefore the expression $\rho_3 = P_3 P_4/(Q_2 Q_3 + P_3 P_4)$ is just the success probability over the probability space.

Lemma 4 *Consider any trajectory-based algorithm that fits in Algorithm* 1 *on* $f_5^{M,a,b,M'}$ *starting from the node* 3. *Then the absorbing probability for state* 5 *is*

$$\rho_3 = \frac{P_3 P_4}{Q_2 Q_3 + P_3 P_4}.$$

Proof Firstly we compute the absorbing probabilities,

$$\begin{aligned}
&\rho_1 = 0\\
&\rho_2 = p_2\rho_3 + q_2\rho_1 + (1 - p_2 - q_2)\rho_2\\
&\rho_3 = p_3\rho_4 + q_3\rho_2 + (1 - p_3 - q_3)\rho_3\\
&\rho_4 = p_4\rho_5 + q_4\rho_3 + (1 - p_4 - q_4)\rho_4\\
&\rho_5 = 1
\end{aligned}$$

which can be rewritten using P_i and Q_i from Eq. (3) and the two boundary conditions as

$$\begin{aligned}\rho_2 &= P_2\rho_3\\ \rho_3 &= P_3\rho_4 + Q_3\rho_2\\ \rho_4 &= P_4 + Q_4\rho_3.\end{aligned}$$

Solving the previous system for ρ_3 yields $\rho_3 = P_3 \cdot (P_4 + Q_4\rho_3) + Q_3 P_2 \rho_3$ which leads to

$$\rho_3 = \frac{P_3 P_4}{1 - Q_3 P_2 - P_3 Q_4}.$$

Introducing $Q_3 = 1 - P_3$, $P_2 = 1 - Q_2$ and $Q_4 = 1 - P_4$ in the denominator yields the claimed statement. □

Now we apply the previous general result for the two studied heuristics. First, for Metropolis one would expect the absorbing probability to be $1/2$ since it does not distinguish between improving moves of different magnitudes. However, it comes as a surprise that this probability will always be $> 1/2$. The reason is again due to the fitness dependent acceptance probability of detrimental moves.

Theorem 5 *Consider MA starting from state* 3 *on* $f_5^{M,a,b,M'}$. *Then the absorbing probability for state* 5 *is*

$$\rho_3^{\text{MA}} = \frac{1 + e^{-\alpha a}}{2 + e^{-\alpha a} + e^{-\alpha b}} > \frac{1}{2}.$$

Proof First let us compute the two conditional probabilities

$$Q_2 = \frac{1}{1 + e^{-\alpha a}}, \qquad P_4 = \frac{1}{1 + e^{-\alpha b}}.$$

Now we invoke Lemma 4 but with $P_3 = Q_3 = 1/2$ since Metropolis does not distinguish slope gradients. Hence,

$$\rho_3 = \frac{P_4}{Q_2 + P_4} = \frac{1/\left(1 + e^{-\alpha b}\right)}{1/\left(1 + e^{-\alpha a}\right) + 1/\left(1 + e^{-\alpha b}\right)} = \frac{1 + e^{-\alpha a}}{2 + e^{-\alpha a} + e^{-\alpha b}}.$$

Finally, using $a < b$, it follows that $\rho_3^{\text{MA}} > 1/2$. □

Finally, for SSWM we were able to reduce the complexity of the absorbing probability to just the two intermediate points (states 2 and 4) between the valley (state 3) and the two optima (states 1 and 5). The obtained expression is reminiscent of the absorbing probability on the 3 State Model (Theorem 2). However, it is important to note that a and b were the fitness of the optima in $f_3^{a,b}$ and now they refer to the transition nodes between the valley and the optima.

Theorem 6 *Consider SSWM ($N \geq 2$) starting from state 3 on $f_5^{M,a,b,M'}$. Then the absorbing probability of state 5 is*

$$\rho_3^{\text{SSWM}} \geq \frac{p_{\text{fix}}(b)}{p_{\text{fix}}(b) + p_{\text{fix}}(a)} > \frac{1}{2}.$$

Proof Let us start by computing the probabilities required by Lemma 4.

$$P_4 = \frac{1}{1 + p_{\text{fix}}(-b)/p_{\text{fix}}(M' - b)} \quad Q_2 = \frac{1}{1 + p_{\text{fix}}(-a)/p_{\text{fix}}(M - a)}$$
$$P_3 = \frac{1}{1 + p_{\text{fix}}(a)/p_{\text{fix}}(b)} \qquad Q_3 = \frac{1}{1 + p_{\text{fix}}(b)/p_{\text{fix}}(a)}$$

Let us now focus on the term $Q_2 Q_3/(P_3 P4)$:

$$\frac{Q_2 Q_3}{P_3 P_4} = \frac{\left(1 + \frac{p_{\text{fix}}(-b)}{p_{\text{fix}}(M'-b)}\right)}{\left(1 + \frac{p_{\text{fix}}(-a)}{p_{\text{fix}}(M-a)}\right)} \cdot \frac{\left(1 + \frac{p_{\text{fix}}(a)}{p_{\text{fix}}(b)}\right)}{\left(1 + \frac{p_{\text{fix}}(b)}{p_{\text{fix}}(a)}\right)}$$

the last term is of the form $(1 + x)/(1 + 1/x) = x$, hence it can be highly simplified to just $p_{\text{fix}}(a)/p_{\text{fix}}(b)$, yielding

$$\frac{Q_2 Q_3}{P_3 P_4} = \frac{\left(1 + \frac{p_{\text{fix}}(-b)}{p_{\text{fix}}(M'-b)}\right)}{\left(1 + \frac{p_{\text{fix}}(-a)}{p_{\text{fix}}(M-a)}\right)} \cdot \frac{p_{\text{fix}}(a)}{p_{\text{fix}}(b)}$$

since $0 < p_{\text{fix}}(-b) < p_{\text{fix}}(-a) < p_{\text{fix}}(M - a) < p_{\text{fix}}(M' - b) < 1$, we can bound $p_{\text{fix}}(-b)/p_{\text{fix}}(M' - b) \leq p_{\text{fix}}(-a)/p_{\text{fix}}(M - a)$ to obtain

$$\frac{Q_2 Q_3}{P_3 P_4} \leq \frac{\left(1 + \frac{p_{\text{fix}}(-a)}{p_{\text{fix}}(M-a)}\right)}{\left(1 + \frac{p_{\text{fix}}(-a)}{p_{\text{fix}}(M-a)}\right)} \cdot \frac{p_{\text{fix}}(a)}{p_{\text{fix}}(b)} = \frac{p_{\text{fix}}(a)}{p_{\text{fix}}(b)}.$$

Substituting this in Lemma 4 leads to

$$\rho_3 = \frac{1}{1 + Q_2 Q_3/(P_3 P_4)} \geq \frac{1}{1 + p_{\text{fix}}(a)/p_{\text{fix}}(b)} = \frac{p_{\text{fix}}(b)}{p_{\text{fix}}(b) + p_{\text{fix}}(a)}.$$

Finally, using $b > a$ we obtain the lower bound of $1/2$. □

5.1 An Example Where SSWM Outperforms Metropolis

We now consider a smaller family of problems $f_5^{M,1,10,M'}$ and create an example where SSWM outperforms Metropolis. In this simpler yet general scenario we can compute the optimal temperature for Metropolis that will maximise the absorbing probability ρ_3^{MA}.

Lemma 7 *Consider Metropolis on $f_5^{M,1,10,M'}$ starting from state 3. Then for any parameter $\alpha \in \mathbb{R}^+$ the absorbing probability ρ_3^{MA} of state 5 can be bounded as*

$$\rho_3^{\mathrm{MA}}(\alpha) \leq \rho_3^{\mathrm{MA}}(\alpha^*) < 0.63$$

where $\alpha^* = 0.312\ldots$ is the optimal value of α.

Proof Introducing the problem settings ($a = 1$ and $b = 10$) in the absorbing probability from Theorem 5 yields

$$\rho_3^{\mathrm{MA}}(\alpha) = \frac{1 + e^{-\alpha}}{2 + e^{-\alpha} + e^{-10\alpha}}$$

whose derivative is

$$\frac{d\rho_3^{\mathrm{MA}}(\alpha)}{d\alpha} = \frac{e^{9\alpha}\left(10e^{\alpha} - e^{10\alpha} + 9\right)}{\left(e^{9\alpha} + 2e^{10\alpha} + 1\right)^2}.$$

By solving numerically this equation for $d(\rho_3^{\mathrm{MA}}(\alpha))/d\alpha = 0$ with $\alpha > 0$ we obtain an optimal value of $\alpha^* = 0.312071\ldots$ which yields the maximum value of $\rho_3^{\mathrm{MA}}(\alpha^*) = 0.623881\ldots$ (see Fig. 5). □

Now that we have shown the optimal parameter for Metropolis, we will find parameters such that SSWM outperforms Metropolis. To obtain this we must make use of SSWM's ability of rejecting improvements. We wish to identify a parameter setting such that small improvements ($\Delta f = a = 1$) are accepted with small probabilities, while large improvements ($\Delta f = b = 10$) are accepted with a considerably higher probability. The following graph shows p_{fix} for different values of β. While for large β, $p_{\mathrm{fix}}(1)$ and $p_{\mathrm{fix}}(10)$ are similar, for smaller values of β there is a significant difference.

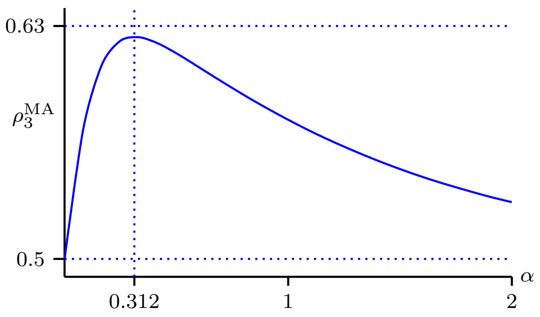


Fig. 5 Absorbing probability of Metropolis on the 5-state model

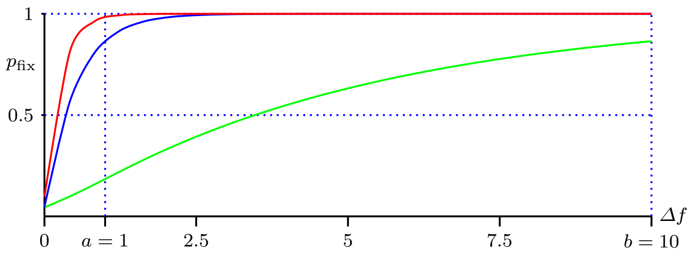


Fig. 6 Acceptance probability of SSWM with $N = 20$ and $\beta = (0.2\,,\,2\,,\,4)$ for the (green, blue, red) curves (Color figure online)

Furthermore we can see that $p_{\text{fix}}(1) \leq 1/2$ i.e. the algorithm will prefer to stay in the current point, rather than moving to the local optimum.

In the following lemma we identify a range of parameters for which the desired effect occurs. The results hold for arbitrary population size, apart from the limit case $N = 1$ where SSWM becomes a pure random walk. The scaling factor β is the crucial parameter; only small values up to 0.33 will give a better performance than Metropolis.

Lemma 8 *Consider SSWM on $f_5^{M,1,10,M'}$ starting from state* 3. *Then for* $\beta \in (0, 0.33]$ *and* $N \geq 2$ *the absorbing probability* ρ_3^{SSWM} *of state* 5 *is at least* 0.64.

Proof Using the bound on ρ_3^{SSWM} from Theorem 6 with $a = 1$ and $b = 10$ we obtain

$$\rho_3^{\text{SSWM}} \geq \frac{p_{\text{fix}}(10)}{p_{\text{fix}}(1) + p_{\text{fix}}(10)} = \frac{1}{1 + p_{\text{fix}}(1)/p_{\text{fix}}(10)}.$$

We want to show that $\rho_3^{\text{SSWM}} \geq 0.64$, which is equivalent to $p_{\text{fix}}(1)/p_{\text{fix}}(10) \leq 1/0.64 - 1 = 9/16$. For that, we use the following bounds from Lemma 1 in [15]: for all $\Delta_f > 0$,

$$\frac{2\beta\Delta f}{1 + 2\beta\Delta f} \leq p_{\text{fix}}(\Delta f) \leq \frac{2\beta\Delta f}{1 - e^{-2N\beta\Delta f}}.$$

Using these two inequalities for $\Delta f = 1$ and $\Delta f = 10$ respectively, we obtain

$$\frac{p_{\text{fix}}(1)}{p_{\text{fix}}(10)} \leq \frac{2\beta}{1 - e^{-2N\beta}} \cdot \frac{1 + 20\beta}{20\beta} = \frac{1 + 20\beta}{10\left(1 - e^{-2N\beta}\right)} \leq \frac{1 + 20\beta}{10\left(1 - e^{-4\beta}\right)},$$

where in the last step we have used $N \geq 2$. The obtained expression is always increasing with $\beta > 0$, hence we just need to find the value β^* for when it crosses our threshold value of $9/16$. Solving this numerically we found that the value is $\beta^* = 0.332423\ldots$, and the statement will be true for β values up to this cut off point (see Fig. 6). $\square$

Now that we have derived parameter values for which SSWM has a higher absorbing probability on the 5 state model than Metropolis for any temperature setting $1/\alpha$ (Lemma 7), we are ready to construct a function where SSWM considerably outperforms Metropolis. We first define a concatenated function

$$f(X) = \sum_{i=1}^{n} f_5^{M,a,b,M'}(x_i)$$

consisting of n copies of the 5 state model (i.e. n components) x_i with $1 \leq i \leq n$, such that the concatenated function $f(x)$ returns the sum of the fitnesses of the individual components. Note that $3n$ bits are used in total. To ensure that the algorithms take long expected times to escape from each local optimum we set $M = n$ and $M' = 2n$ for each component x_i, apart from keeping $a = 1$ and $b = 10$, for which the absorbing probabilities from Lemmas 7 and 8 hold. Furthermore, we assume $2\beta(N-1) = \Omega(1)$ to ensure that SSWM remains in states 1 or 5 for a long time.

Theorem 9 *The expected time for SSWM and Metropolis to reach either the local or global optimum of all the components of $f_5^{n,1,10,2n}$ is $O(n \log n)$. With overwhelming probability $1 - e^{-\Omega(n)}$, SSWM with positive constant $\beta < 0.33$ and $N \geq 2$ has optimised correctly at least $(639/1000)n$ components while Metropolis with optimal parameter $\alpha = 0.312\ldots$ has optimised correctly at most $(631/1000)n$ components. The expected time for either algorithm to increase (or decrease) further the number of correctly optimised components by one is at least $e^{\Omega(n)}$.*

Proof The expected time to reach either of the states 5 or 1 on the single-component 5 state model is a constant c for both algorithms. Hence, the first statement follows from an application of the coupon collector where each coupon has to be collected c times [12]. The second statement follows by straightforward applications of Chernoff bounds using that each component is independent and, pessimistically, that SSWM optimises each one correctly with probability $640/1000$ (i.e., Lemma 8) and Metropolis with probability $630/1000$ (i.e., Lemma 7). The final statement follows because both algorithms with parameters $\Omega(1)$ accept a new solution, that is $\Omega(n)$ worse, only with exponentially small probability. $\square$

As the absorbing probabilities of SSWM and Metropolis are both constants, with that of SSWM being higher than that of MA, we expect SSWM to achieve a higher fitness. We can amplify these potentially small differences by defining an indicator function returning 1 if at least a certain number of components are optimised correctly (i.e. state 110 is found) and 0 otherwise:

$$g(X) := \begin{cases} 1 & \text{if at least } 0.635n \text{ components are in the global optimum state} \\ 0 & \text{otherwise.} \end{cases}$$

We use this to compose a function h where with overwhelming probability SSWM is efficient while Metropolis is not:

$$h(X) = f(X) \cdot (1 - g(X)) + 2nM' \cdot g(X)$$

Note that $h(X) = f(X)$ while the indicator function $g(X)$ returns 0, and h attains a global optimum if and only if $g(X) = 1$. Our analysis transfers to the former case.

Corollary 10 *In the setting described in Theorem 9, with probability $1 - e^{-\Omega(n)}$ SSWM finds an optimum on $h(X)$ after reaching either the local or global optimum on every component (which happens in expected time $O(n \log n)$), while Metropolis requires $e^{\Omega(n)}$ steps with probability $1 - e^{-\Omega(n)}$.*

Obviously, by swapping the values of M and M' in f, the function would change into one where preferring improvements of higher fitness is deceiving. As a result, SSWM would, with overwhelming probability, optimise at least 63.9% of the components incorrectly. Although Metropolis would optimise more components correctly than SSWM, it would still be inefficient on h.

5.2 Experiments

We performed experiments to study the performance of SSWM and MA on the 5 state model under several parameter settings. The experimental setting is similar to that of the 3 state model. We can see in Fig. 7 how: while SSWM is able to reach the performance threshold imposed by $g(X)$, MA is not. As expected, both algorithms start with a g-value of 0 and hence they are optimising $f(X)$. However, for SSWM, once the dashed line on Fig. 7 is reached, $g(X)$ suddenly changes to 1 and $h(X)$ is optimised, hence the flat effect on SSWM's curves.

We also plot the indicator function $g(X)$ as this is the most crucial term in $h(X)$. Again the results from Fig. 8 are in concordance with the theory showing that SSWM outperforms MA. However, we observe that when choosing effective values of the temperature ($\alpha = 0.18$ in the figure) we can see that a small fraction of runs of MA manage to optimise $g(X)$ yielding a non-zero expected value. The opposite effect can be seen for SSWM on the green curve, although its average g-value is much better than MA's, not all the runs made it to $g(X) = 1$. We believe that this is because the chosen problem size is not large enough. If we recall Theorem 9, MA will in expectation optimise up to $(631/100)n$ components and SSWM at least $(639/1000)n$. This means that the gap for our chosen value of $n = 500$ is just 4 components, which can be achieved by some runs deviating from the expected behaviour. Due to limited computational resources we were unable to consider larger values of n.

6 When is it Beneficial to Exploit?

We further analyse the performance of other common single-trajectory-based search algorithms on the function classes we identified in the previous sections. The reason that SSWM outperforms Metropolis for the identified composite function is that the former algorithm tends to favour the acceptance of search points on the slope of largest uphill gradient while the latter algorithm accepts any improvement independent of its quality. Hence, we expect that also other algorithms that prefer improvements of higher quality over smaller ones (i.e., a characteristic often referred to as *exploitation*) to also perform well on the composite function. A well known algorithm that prefers exploitation is the traditional local search strategy that selects the best improvement in the neighbourhood of the current search point, that is, Best-Improvement Local Search (BILS). In particular, since a similar distinction between the behaviours of SSWM and Metropolis is also present between BILS and the local search strategy which selects the first found improvement, that is, First Improvement Local Search (FILS) in the current neighbourhood, we will analyse the performance of these two algorithms. This also relates to previous work where the choice of the pivot rule was investigated in

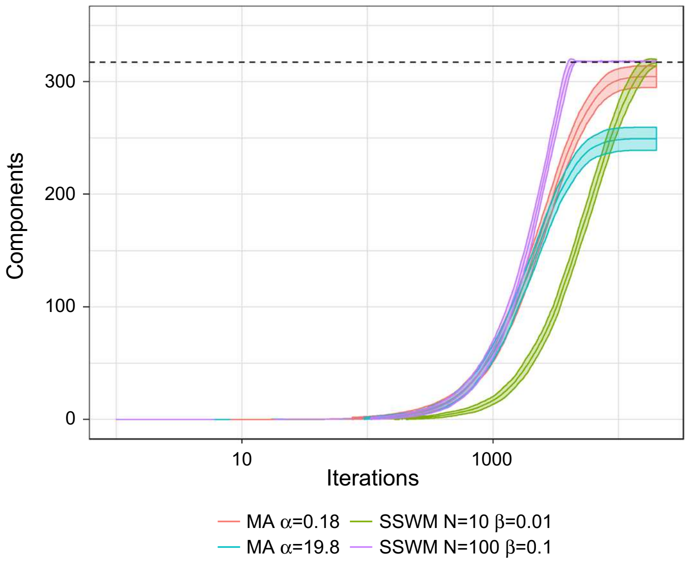


Fig. 7 Average number of components at state 5 over time by SSWM and MA when optimising $h(X)$ with 500 components of the 5 state model. For Metropolis the temperature was chosen such that $\alpha = 2(N-1)\beta$. Results are averaged over 50 independent runs and the shadowed zones include $\pm$ one standard deviation. A logarithmic scale with base 10 is used for the x-axis. The dashed line ($y = 500 * 0.635$) indicates the threshold established in the definition of the step function $g(X)$

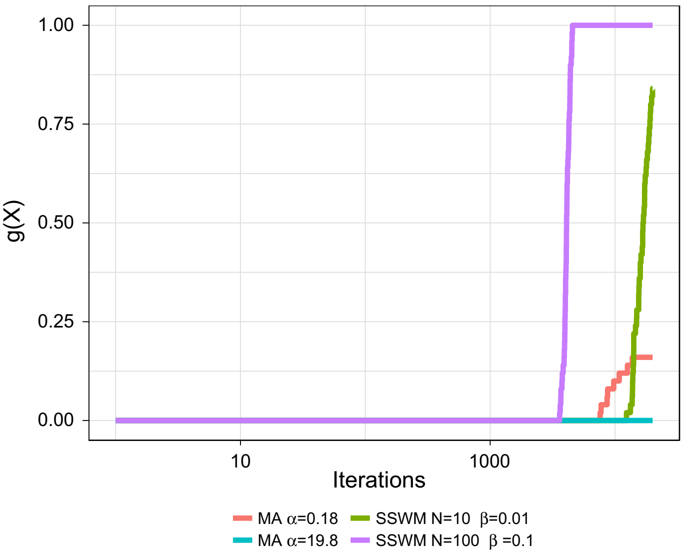


Fig. 8 Average $g(X)$ values over time for SSWM and MA when optimising $h(X)$ with 500 components of the 5 state model. For Metropolis the temperature was chosen such that $\alpha = 2(N-1)\beta$. Results are averaged over 50 independent runs and a logarithmic scale with base 10 is used for the x-axis

local search and memetic algorithms that combine evolutionary algorithms with local search [4, 19, 22].

The pseudo-code for FILS and BILS are respectively given in Algorithms 2 and 3 (see e.g. [22]). These two optimisers, like any Algorithm 1 with local mutations, can only explore the Hamming neighbourhood in one iteration. FILS will keep producing distinct Hamming neighbours until it finds an improvement, whilst BILS computes the set of all neighbours and chooses one of those with the highest fitness. Both algorithms stop when there is no improving neighbour.

Algorithm 2 FILS (Adapted from Algorithm 4 in [22])

```
Initialise $x \in \{0, 1\}^n$
$i \leftarrow 0$
**repeat**
    Generate a random permutation Per of length $n$
    **for** $i = 1$ to $n$ **do**
        $y \leftarrow$ flip the Per[$i$]-th bit of $x$
        **if** $f(y) > f(x)$ **then**
            $x \leftarrow y$
            go to line 4
        **end if**
    **end for**
    stop
**until** stop
```

Algorithm 3 BILS (Adapted from Algorithm 3 in [22])

```
Initialise $x \in \{0, 1\}^n$
**repeat**
    BestNeighbourSet $= \emptyset$
    **for** $i = 1$ to $n$ **do**
        $y \leftarrow$ flip the $i$-th bit of $x$
        **if** $f(y) > f(x)$ **then**
            BestNeighbourSet $=$ BestNeighbourSet $\cup\, y$
        **end if**
    **end for**
    **if** BestNeighbourSet $= \emptyset$ **then**
        stop
    **end if**
    $x$ is uniform randomly chosen from $\arg\max$ (BestNeighbourSet)
**until** stop
```

We will also consider a classical single trajectory evolutionary algorithm that favours exploitation. In order to achieve a fair performance comparison with SSWM and Metropolis we consider the $(1,\lambda)$ RLS algorithm which, like the former algorithms, uses non-elitism and local mutations. The algorithm creates λ new solutions, called *offspring*, at each step by mutating the current search point, and then it selects the best offspring, independent of whether it is an improvement. If the number of offspring λ

is sufficiently large, then with high probability the slope with steepest gradient will be identified on one component.

The pseudo-code of the (1,λ) RLS is given in Algorithm 4. This optimiser produces λ offspring by flipping one bit chosen uniformly at random independently for each offspring, and then choosing a best one to survive to the next generation. Although the selection mechanism picks the best offspring for survival, the (1,λ) RLS is not an elitist algorithm. Since the parent genotype is left out of the fitness comparison, if the λ children have a lower fitness than the current solution, then the algorithm will move to a search point of lower fitness.

Algorithm 4 (1,λ) RLS

```
Initialise $x \in \{0,1\}^n$
repeat
  for $i = 1$ to $\lambda$ do
    $y_i \leftarrow$ flip uniformly at random one bit from $x$
  end for
  $x \leftarrow$ uniform randomly chosen from $\arg\max(f(y_1), f(y_2), \ldots, f(y_\lambda))$
until stop
```

6.1 Analysis for the 3 State Model

We first derive the absorbing probabilities of the three algorithms introduced in Sect. 6 on the 3 state model. Theorem 11 confirms that BILS optimises the 2-bit function with probability 1 while FILS only does so with probability 1/2. On the other hand, Theorem 12 reveals that the (1,λ) RLS always outperforms FILS for any $\lambda > 1$ and converges to the performance of BILS as the offspring population size λ increases.

Theorem 11 *Consider FILS and BILS on $f_3^{a,b}$ starting from state 2. Then the absorbing probabilities of state 3, respectively, are*

$$\rho_2^{\text{FILS}} = \frac{1}{2} \quad \text{and} \quad \rho_2^{\text{BILS}} = 1.$$

Proof FILS will produce either state 1 or state 3 (both with probability 1/2) and accept the fitness change. Hence, like Metropolis, FILS has transition probabilities $p_2 = q_2$ which, after a direct application of Theorem 2, yields the claimed result.

On the other hand, BILS will produce both state 1 and state 3, and move to the latter since it has higher fitness. Hence, $q_2 = 0$ and $p_2 = 1$ which leads to an absorbing probability of 1 by Theorem 2. □

Theorem 12 *Consider the (1,λ) RLS on $f_3^{a,b}$ starting from state 2. Then, the absorbing probability of state 3 is*

$$\rho_2^{(1,\lambda)\,RLS} = 1 - 2^{-\lambda}.$$

Proof In order for the (1,λ) RLS to move to state 3 from state 2 it suffices to create just one offspring at state 3 (the global optimum). The probability of creating such a search point is just the probability of choosing the first bit to be flipped, which is $1/2$. Then, with probability $(1-1/2)^\lambda = 2^{-\lambda}$ none of the λ offspring will be at state 3. And, the probability of at least one child being at the global optimum is $1-2^{-\lambda}$.

Hence, $p_2 = 1-2^{-\lambda}$ and since every mutation of state 2 leads to either state 1 or state 3, $q_2 = 1-p_2 = 2^{-\lambda}$. Introducing this in Theorem 2 we obtain $\rho_2 = p_2$. □

6.2 Analysis for the 5 State Model

We now derive the absorbing probabilities of the three algorithms for the 5 state model. The absorbing probabilities for BILS and FILS as stated in the theorem below are the same as for the 3 state model.

Theorem 13 *Consider FILS and BILS on* $f_5^{M,a,b,M'}$ *starting from state* 3. *Then the absorbing probabilities of state* 5, *respectively, are*

$$\rho_3^{\text{FILS}} = \frac{1}{2} \qquad \textit{and} \qquad \rho_3^{\text{BILS}} = 1.$$

Proof For FILS, a direct application of Lemma 4 with $P_4 = 1$, $P_3 = 1/2$, $Q_2 = 1$ and $Q_3 = 1/2$ yields an absorbing probability of $1/2$.

For BILS, Lemma 4 with $P_4 = 1$, $P_3 = 1$, $Q_2 = 1$ and $Q_3 = 0$ yields an absorbing probability of 1. □

Interestingly, the analysis of (1,λ) RLS on the 5 state model turns out to be more complex than that of SSWM, Metropolis, and (1,λ) RLS on the 3 state model as for the 5 state model it is possible for the algorithm to reach search points of fitness $-\infty$. This is because the non-absorbing states have Hamming neighbours of fitness $-\infty$, and such a search point is reached in case all λ offspring happen to have this fitness. While the genotypic encoding was irrelevant in all previous settings, it does become relevant in the following analysis.

Theorem 14 shows that the absorbing probability of the (1,λ) RLS converges to 1 slightly more slowly as λ increases than the one derived for the 3 state model.

Theorem 14 *Consider the (1,λ) RLS starting from state* 3 *on* $f_5^{M,a,b,M'}$. *Then the absorbing probability of state* 5 *is*

$$\rho_3^{\textit{(1,}\lambda\textit{) RLS}} = \frac{1-(2/3)^\lambda}{1-(1/3)^\lambda}.$$

Proof Since the (1,λ) RLS can move to states with a fitness of $-\infty$, the diagram from Fig. 4 is incomplete. However, let us focus now on the Hamming neighbours of each state. Recall that our genotype encoding of the 5 state model is based on 3 bits. We observe that, apart from the two maximal states (states 1 and 5), the three neighbours of each state have mutually different fitness values. Hence, we denote by p, q and r the transition probabilities towards the neighbour with the highest, intermediate and lowest fitness, respectively. Using this notation, we can express the absorbing probabilities as

$$\begin{aligned}
\rho_1 &= 0\\
\rho_2 &= q\rho_3 + r\rho_7\\
\rho_3 &= q\rho_2 + p\rho_4 + r\rho_6\\
\rho_4 &= q\rho_3 + p + r\rho_7\\
\rho_5 &= 1\\
\rho_6 &= r\rho_3 + p\\
\rho_7 &= q\rho_2 + p\rho_4 + r\rho_8\\
\rho_8 &= p + r\rho_7.
\end{aligned}$$

We now move to a matrix formulation of the form $A\boldsymbol{\rho} = \boldsymbol{b}$. But first, we plug in ρ_8 in ρ_7 and we no longer consider the trivial $\rho_1 = 0$ and $\rho_5 = 1$, hence $\boldsymbol{\rho} = (\rho_2, \rho_3, \rho_4, \rho_6, \rho_7)^\top$, leading to

$$\begin{pmatrix} 1 & -q & 0 & 0 & -r\\ -q & 1 & -p & -r & 0\\ 0 & -q & 1 & 0 & -r\\ 0 & -r & 0 & 1 & 0\\ -q & 0 & -p & 0 & 1-r^2 \end{pmatrix} \cdot \begin{pmatrix} \rho_2\\ \rho_3\\ \rho_4\\ \rho_6\\ \rho_7 \end{pmatrix} = \begin{pmatrix} 0\\ 0\\ p\\ p\\ rp \end{pmatrix}.$$

The solution will be $\boldsymbol{\rho} = A^{-1}\boldsymbol{b}$, but we are just interested in ρ_3. Then, taking the second row of A^{-1} (here denoted as $\boldsymbol{A}_2^{-1}$) we can express the absorbing probability as $\rho_3 = \boldsymbol{A}_2^{-1}\boldsymbol{b}$. By standard matrix calculations, we obtain

$$\boldsymbol{A}_2^{-1} = \frac{1}{(p+r)(1-r)} \cdot \left(q,\ \tfrac{1}{1+r},\ p,\ \tfrac{r}{1+r},\ \tfrac{r}{1+r}\right),$$

which can be verified with the expression $A^\top \left(\boldsymbol{A}_2^{-1}\right)^\top = (0, 1, 0, 0, 0)$. Finally, we compute $\rho_3 = \boldsymbol{A}_2^{-1}\boldsymbol{b}$ as follows:

$$\begin{aligned}
\rho_3 &= \frac{1}{(p+r)(1-r)} \cdot \left(1-p-r,\ \tfrac{1}{1+r},\ p,\ \tfrac{r}{1+r},\ \tfrac{r}{1+r}\right) \cdot \begin{pmatrix} 0\\ 0\\ p\\ p\\ rp \end{pmatrix}\\
&= \frac{p^2 + pr/(1+r) + r^2p/(1+r)}{(p+r)(1-r)}\\
&= \frac{p^2(1+r) + pr + r^2p}{(p+r)(1-r)(1+r)}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{p^2(1+r) + pr(1+r)}{(p+r)(1-r)(1+r)} \\
&= \frac{p^2 + pr}{(p+r)(1-r)} \\
&= \frac{p(p+r)}{(p+r)(1-r)} \\
&= \frac{p}{1-r}. \qquad (4)
\end{aligned}
$$

Finally, we just have to introduce the values of p and r. First, to move to the neighbour with the highest fitness, it is sufficient to produce one offspring at the desired search point. Noticing that $(1-1/3)^\lambda$ is the probability that none of the offspring are at the best neighbour, it follows that $p = 1-(1-1/3)^\lambda = 1-(2/3)^\lambda$. In order to move to the neighbour with the lowest fitness, all λ offspring must be equal to said neighbour, which happens with probability $r = (1/3)^\lambda$. Introducing these values in Eq. (4) leads to the claimed statement. $\square$

Introducing $\lambda \geq 3$ in the expression obtained in Theorem 14, which is monotonically non-decreasing with λ, leads to

$$\rho_3^{(1,\lambda)\text{ RLS}} \geq \frac{1-(2/3)^3}{1-(1/3)^3} = \frac{1-8/27}{1-1/27} = \frac{19}{26} = 0.7307\cdots \geq 0.64.$$

Hence already an offspring population size of $\lambda = 3$ is sufficient to raise the success probability above that of the Metropolis algorithm with optimal parameters.

However, it is not straightforward to translate our results from one component $f_5^{M,a,b,M'}$ to n components. Unlike for SSWM and Metropolis, on $n \gg 1$ components the (1,λ) RLS is likely to perform mutations in different components. Our analysis from Theorem 14 breaks down as all transition probabilities rely on the fact that all λ mutations concern the same component.

The dynamics on $n \gg 1$ components seem very different to the dynamics on one component, and quite complex. We therefore resort to experiments to shed light on the performance of (1,λ) RLS on n components and our composite function h.

6.3 Experiments

We present experimental results to understand the dynamics of the (1,λ) RLS on concatenated components of the 5 state model. Figure 9 shows the behaviour of the (1,λ) RLS when optimising $f(X)$ with 100 components. It is important to note that this setting does not exactly match the one from Fig. 7, as there the algorithms were optimising the function $h(X)$. The only difference is that in Fig. 9 the algorithms can keep optimising components once the dashed line ($g(X) = 1$) is reached.

We observe an interesting effect for small values of λ. The algorithm starts accumulating components at state 5, however, at some point in time, the fitness decreases to that of a random configuration. This is due to the fact that states 6, 7 and 8 have

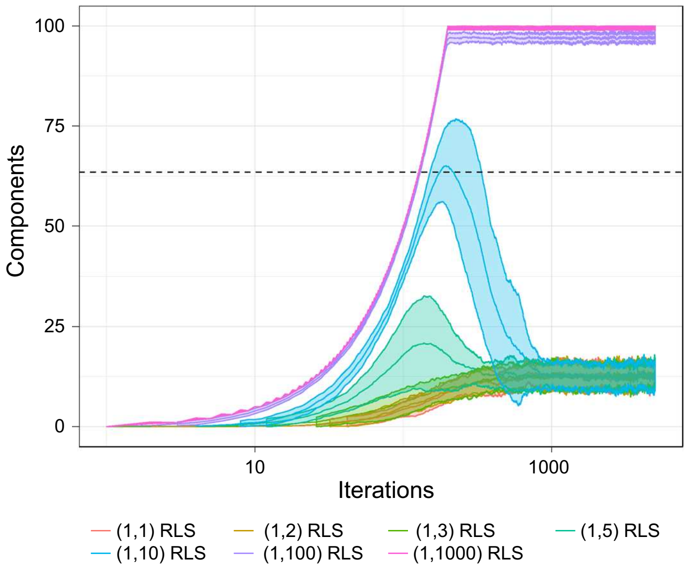


Fig. 9 Average number of components correctly optimised over time by the (1,λ) RLS on 100 concatenated components of the 5 state model. Results are averaged over 50 independent runs and the shadowed zones include $\pm$ one standard deviation. A logarithmic scale with base 10 is used for the x-axis. The dashed line ($y = 63.5$) indicates the threshold established on the definition of the step function $g(X)$

a value of $-\infty$ for $f_5^{M,a,b,M'}$. If at some point in time, the algorithm sets just one component to either of these states, the total fitness $f(X)$ will be $-\infty$, no matter the fitness of the remaining components. Then, all that the (1,λ) RLS sees are points of equal fitness and it just chooses one uniformly at random. Obviously, the larger the λ, the smaller the probability of sampling a point with $f(X) = -\infty$ in the first place and therefore, as seen in the figure, large values of λ manage to reach the threshold imposed by $g(X)$.

We now move to the study of the (1,λ) RLS when optimising $h(X)$. This is shown in Fig. 10 by plotting the step function $g(X)$ as this is the most crucial term in $h(X)$. As suggested by Fig. 9, a sufficiently large value of λ is needed to ensure that all runs optimise $g(x)$ and thus $h(X)$.

We conclude the subsection by presenting in Fig. 11 a comparison graph that plots the performance of all the algorithms considered in this chapter. While BILS optimises all the components, the performance of SSWM and the (1,λ) RLS is comparable and outperform the other algorithms. In particular, they both identify correctly a sufficient number of components such that they find the optimum of the composite function h.

7 Conclusions and Future Work

We have presented a rigorous comparison of the non-elitist SSWM and Metropolis algorithms. Their main difference is that SSWM may reject improving solutions while Metropolis always accepts them. Nevertheless, we prove that both algorithms

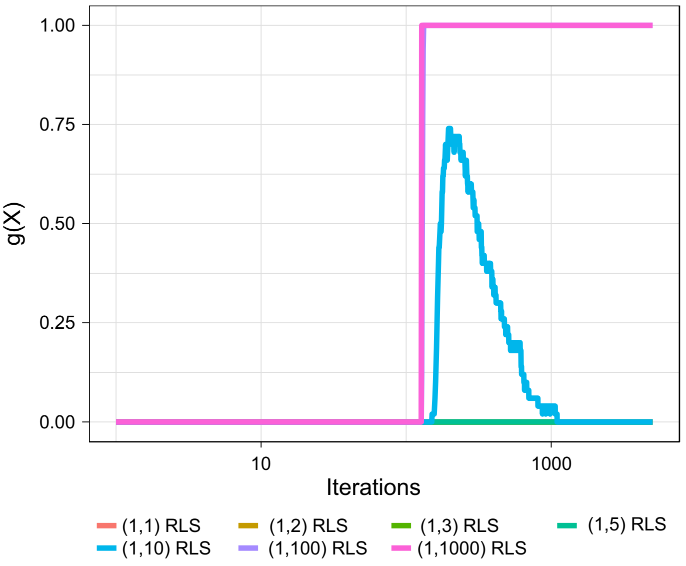


Fig. 10 Average $g(X)$ values over time for the (1,λ) RLS when optimising $h(X)$ for 100 components of the 5 state model. Results are averaged over 50 independent runs and a logarithmic scale with base 10 is used for the x-axis. Note that the (1,λ) RLS with $\lambda \leq 5$ always has a value of 0 and the (1, 100) RLS is covered by the results of the (1, 1000) RLS

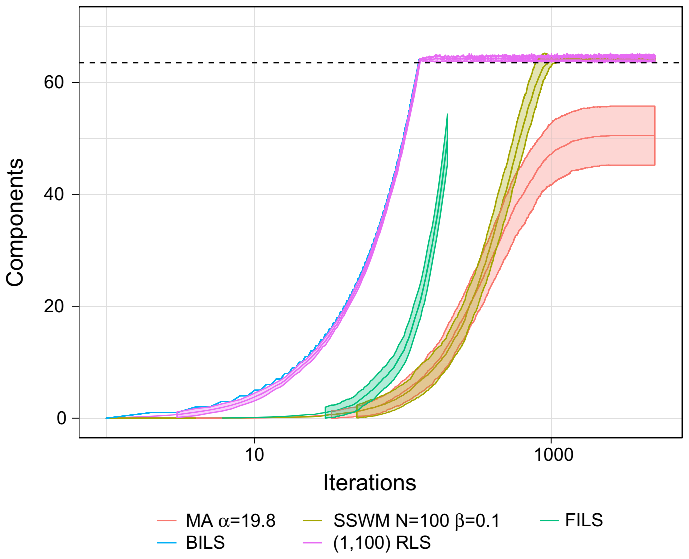


Fig. 11 Average number of components correctly optimised over time by all the algorithms when optimising $h(X)$ with 100 concatenated components of the 5 state model. Results are averaged over 50 independent runs and the shadowed zone includes ± one standard deviation. A logarithmic scale with base 10 is used for the x-axis. The dashed line ($y = 63.5$) indicates the threshold established on the definition of the step function $g(X)$. Note that the curve for BILS is mainly covered by the curve for the (1100) RLS. Recall that BILS and FILS stop in local optima, hence the respective curves may finish early

have the same stationary distribution, and they may only have considerably different performance on optimisation functions where the mixing time is large.

Our analysis on a 3 state model highlights that a simple function with a local optimum of low fitness and a global optimum of high fitness does not allow the required large mixing times. The reason is that, although Metropolis initially chooses the local optimum more often than SSWM, it still escapes quickly. As a result we designed a 5 state model which "locks" the algorithms to their initial choices. By amplifying the function to contain several copies of the 5 state model we achieve our goal of defining a composite function where SSWM is efficient while Metropolis requires exponential time with overwhelming probability, independent from its temperature parameter.

Given the similarities between SSWM and other particularly selective strategies such as steepest ascent and single-trajectory algorithms using offspring populations, we compared the performance of SSWM and Metropolis with BILS, FILS and a (1,λ) RLS. We rigorously showed that BILS excels on the composite function and experiments have shown that the (1,λ) RLS performs comparable to SSWM for large enough λ.

Our theoretical and experimental analyses indicate that SSWM and Metropolis differ in performance in the 'non-elitist world' in a similar way to how Best-Improvement and First Improvement local search (resp. BILS and FILS) differ in the 'elitist world'. In particular, BILS should be preferred if greedy choices (i.e., choosing the locally more promising slope with steepest gradient) are going to be beneficial in the long term compared to taking any improvement (i.e., not necessarily the slope with steepest gradient). If this is not the case, then FILS should be preferred. Our analysis indicates that on problems where BILS outperforms FILS, SSWM will outperform Metropolis (and vice versa). Obviously, for problems where the greedy choice is always the best one throughout the run, then BILS should be preferred to SSWM. However, for problems where the greedy choice is often the best move, but not always, then our analysis suggests that SSWM may perform better than BILS, FILS and Metropolis. We leave to future work an extensive analysis of these conclusions for a wide range of problems including more realistic ones from combinatorial optimisation.

Acknowledgements The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under Grant Agreement No 618091 (SAGE) and from the EPSRC under Grant Agreement No EP/M004252/1. This research is supported by the Basque Government through the BERC 2014–2017 program and by Spanish Ministry of Economy and Competitiveness MINECO through BCAM Severo Ochoa excellence accreditation SEV-2013-0323 and through Project MTM2016-76329-R IMAGEARTH. J.P.H. thanks support from Basque Government - ELKARTEK Programme, grant KK-2016/0002.

References

1. Aldous, D., Fill, J.: Reversible Markov chains and random walks on graphs. Monograph in preparation (2017)

2. Chen, G.-Y., Saloff-Coste, L.: On the mixing time and spectral gap for birth and death chains. Latin Am. J. Probab. Math. Stat. **X**, 293–321 (2013)
3. Feller, W.: An Introduction to Probability Theory and Its Applications. Wiley, Hoboken (1968)
4. Gießen, C.: Hybridizing evolutionary algorithms with opportunistic local search. In: Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation (GECCO ’13), pp. 797–804. ACM (2013)
5. Gillespie, J.H.: Molecular evolution over the mutational landscape. Evolution **38**(5), 1116–1129 (1984)
6. Jerrum, M., Sinclair, A.: The Markov chain Monte Carlo method: an approach to approximate counting and integration. In: Approximation Algorithms for NP-hard Problems, pp. 482–520. PWS Publishing (1996)
7. Kimura, M.: On the probability of fixation of mutant genes in a population. Genetics **47**(6), 713–719 (1962)
8. Levin, D.A., Peres, Y., Wilmer, E.L.: Markov Chains and Mixing Times. American Mathematical Society, Providence (2008)
9. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. **21**(6), 1087–1092 (1953)
10. Nallaperuma, S., Oliveto, P.S., Heredia, J.P., Sudholt, D.: When is it beneficial to reject improvements? In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1391–1398. ACM (2017)
11. Oliveto, P.S., Paixão, T., Pérez Heredia, J., Sudholt, D., Trubenová, B.: When non-elitism outperforms elitism for crossing fitness valleys. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016, GECCO ’16, pp. 1163–1170. New York, NY, USA, ACM (2016)
12. Oliveto, P.S., Yao, X.: Runtime analysis of evolutionary algorithms for discrete optimization. In: Theory of Randomized Search Heuristics: Foundations and Recent Developments, pp. 21–52. World Scientific Publishing Co., Inc. (2011)
13. Orr, H.A.: The population genetics of speciation: the evolution of hybrid incompatibilities. Genetics **139**, 1805–1813 (1995)
14. Paixão, T., Badkobeh, G., Barton, N., Corus, D., Dang, D.-C., Friedrich, T., Lehre, P.K., Sudholt, D., Sutton, A.M., Trubenova, B.: Toward a unifying framework for evolutionary processes. J. Theor. Biol. **383**, 28–43 (2015)
15. Paixão, T., Pérez Heredia, J., Sudholt, D., Trubenová, B.: Towards a runtime comparison of natural and artificial evolution. Algorithmica **78**(2), 681–713 (2017)
16. Pérez Heredia, J., Trubenová, B., Sudholt, D., Paixão, T.: Selection limits to adaptive walks on correlated landscapes. Genetics **205**(2), 803–825 (2017)
17. Sella, G., Hirsh, A.E.: The application of statistical physics to evolutionary biology. Proc. Natl. Acad. Sci. U.S.A. **102**(27), 9541–9546 (2005)
18. Smith, J.E.: Coevolving memetic algorithms: a review and progress report. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) **37**(1), 6–17 (2007)
19. Sudholt, D.: Hybridizing evolutionary algorithms with variable-depth search to overcome local optima. Algorithmica **59**(3), 343–368 (2011)
20. Sudholt, D.: Using Markov-chain mixing time estimates for the analysis of ant colony optimization. In: Proceedings of the 11th Workshop on Foundations of Genetic Algorithms (FOGA 2011), pp. 139–150. ACM Press (2011)
21. Unckless, R.L., Orr, H.A.: Dobzhansky–Muller incompatibilities and adaptation to a shared environment. Heredity **102**(3), 214–217 (2009)
22. Wei, K., Dinneen, M.J.: Runtime analysis to compare best-improvement and first-improvement in memetic algorithms. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO ’14), pp. 1439–1446. ACM (2014)