Studienarbeit

Reducing diversity loss in estimation of distribution algorithms

Universität Karlsruhe (TH)
Fakultät für Informatik?
Institut für Angewandte Informatik und Formale Beschreibungsverfahren

Verantw. Betreuer: Prof. Dr. H. Schmeck
Betr. Mitarbeiter: PD Dr. J. Branke

Beginn: 15.08.2006

Reducing diversity loss in estimation of distribution algorithms

Clemens Lode

Contents

Chapter 1

Reducing diversity loss in EDAs

1.1 Introduction

With inappropriate settings, many EDAs can reach a state from which the probability of ever finding the optimum is zero. This is due to diversity loss which cannot be restored. If any component of the data vectors does not take one of its allowed values anywhere in the entire population, that value can never be restored. If that value is required in the optimum, the optimum will never be sampled. In a flat landscape it was shown that this diversity loss is the same for a whole class of EDAs. A consequence of this is that for a problem which is almost everywhere flat, such as the Needle problem, the probability of diversity loss before the optimum is sampled is also universal for the class. One way to counter this is to increase the population size according to the problem, a different approach is to directly change the distribution vector.

1.2 Abstract

Using the same general class of EDAs as [1] (probability model is build using only data sampled from the last generation), the result that the diversity loss of sampling $\tilde{n}$ individuals from a population is $1 - \frac{1}{\tilde{n}}$ and the calculated result of diversity loss $(1 - \frac{1}{n})$ of generating a population with n individuals from a distribution vector p we come to the conclusion that by changing the distribution vector p accordingly to the population size n and the number of selected individuals $\tilde{n}$, we can reduce the diversity loss. This is true for a flat landscape and tests have shown that while this correction does not outperform a standard Laplace correction in terms of population diversity it works very well for actual non-flat problem like OneMax.
The resulting method is to correct p to $\frac{1}{2}(1 - \sqrt(\frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}))$ for $p < \frac{1}{2}(1 - \sqrt(\frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}))$, to $\frac{1}{2}(1 + \sqrt(\frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}))$ for $p > \frac{1}{2}(1 + \sqrt(\frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}))$ and to $\frac{1}{2}$ else.

1.3 Definitions

The diversity of a given population can be measured by the 'trace of the empirical co-variance matrix'.
Let

- C: number of components of each individual
- n: number of individuals in the population
- $\tilde{n}$: number of selected individuals
- A: set of different values a component can take
- $|A|$: number of different values a component can take

$$x_i^\mu\text{: component i of individual } \mu \tag{1.1}$$

$$v_i^a = \frac{1}{n}\sum_{\mu=1}^{n} \varphi(x_i^\mu = a) \tag{1.2}$$

$$v = \frac{1}{|A|}\sum_{i=1}^{C}\sum_{a=0}^{|A|-1} v_i^a(1-v_i^a) \tag{1.3}$$

Our goal is to determine the resulting diversity of a population created on a given distribution p, so we have to calculate all possible combinations of individuals in that populations of the size n. For simplicity we are looking at strings of the size 1, i.e. $C = 1$, in the case of UMDAs on a flat landscape the results for $C > 1$ are the same, see chapter X.

To create a population with the distribution p each individual has in its component with the probability p a '1' (or with the probability $(1-p)$ a '0'). Creating a population with the size of n with $|A|$ different values in each component we then have a combination with repetition, i.e. there would be

$$\frac{(|A|+n-1)!}{n!(|A|-1)!} = \binom{|A|+n-1}{n} \tag{1.4}$$

possible different population. For example for $|A| = 2$ there would be $n + 1$ different populations (...000, ...001, ...011, ...111, ...) as the position of an individual in the population is not important. In this paper I will only look into the case of $|A| = 2$, i.e. bitsrings. The main difference with $|A| > 2$ compared to $|A| = 2$ is that the probability for a certain population is different and that we no longer can identify a certain population with a 'k', with k being the number of '1's.

The probability for a certain population is

$$p_k = p^k (1-p)^{n-k} \binom{n}{k} \tag{1.5}$$

with k being the number of '1's.

We further define a v_k which represents the diversity for a population with k '1's. The v_i^a values of a v_k with a given k is then:

$$v_i^a(k) = \frac{1}{n} \sum_{\mu=1}^{n} \varphi(x_i^\mu(k) = a) \tag{1.6}$$

As we have defined k as the number of '1's and set $C = 1$ the sum over all $\varphi(x_i^\mu = 1)$ is k (and the sum over all $\varphi(x_i^\mu = 0)$ is $n - k$) and our v_1^a (we only need the v_1^as because we only have one component, i.e. $C = 1$) is:

$$v_1^0 = \frac{n-k}{n}$$

$$v_1^1 = \frac{k}{n}$$

We define the diversity of a given population (with k '1's) as

$$v_k = \frac{1}{2} \sum_{i=1}^{C} \sum_{a=0}^{|A|-1} v_i^a (1 - v_i^a) \tag{1.7}$$

In our case, with bits ($|A| = 2$) and one component ($C = 1$), we get the following:

$$v_k = \frac{1}{2} \sum_{i=1}^{1} \sum_{a=0}^{1} v_i^a (1 - v_i^a) =$$

$$\frac{1}{2}[v_1^0(1 - v_1^0) + v_1^1(1 - v_1^1)] =$$

$$\frac{1}{2}[\frac{n-k}{n}(1 - \frac{n-k}{n}) + \frac{k}{n}(1 - \frac{k}{n})] =$$

$$\frac{kn - k^2}{n^2}$$

Our total diversity d_p for a given distribution p is

$$d_p = \sum_{k=0}^{n} v_k p_k \tag{1.8}$$

What we want is to have a diversity of $p(1 - p)$, exactly the variance of a population of infinite size. What we get is somewhat different. To determine by which factor our result differs from the wanted result, we divide our d_p by $p(1-p)$:

$$\frac{d_p}{p(1-p)} = \sum_{k=0}^{n} \frac{kn-k^2}{n^2} p^k (1-p)^{n-k} \binom{n}{k} =$$

$$\frac{1}{n^2 p(1-p)} \sum_{k=0}^{n} k(n-k) p^k (1-p)^{n-k} \binom{n}{k} =$$

$$\frac{1}{n^2} \sum_{k=1}^{n} k(n-k) p^{k-1} (1-p)^{n-k-1} \binom{n}{k}$$

It can be shown that this can be reduced further. The exact prove will be given elsewhere, here is just the observed result:

$$d = d_p = 1 - \frac{1}{n} \tag{1.9}$$

So it is independent from p, but we still have assumed $|A| = 2$ and $C = 1$. Tests have shown that we get the same result with a random value of C and $|A|$, although the formula in between will be different, especially (4).

According to Shapiro [1] we also know that selecting l individuals from this population of size n will result in a diversity loss (compared to $p(1-p)$) of the same factor $1 - \frac{1}{n}$.

To recap:
We have a distribution p and we generate a new population of the size n. In the optimal case ($n = \inf$) we get the expected variance of $p(1-p)$. We calculated that the real variance is $p(1-p)(1-\frac{1}{n})$ and we know that creating a new population and selecting $\tilde{n}$ individuals from that population (in a flat fitness landscape) will result in the variance $p(1-p)(1-\frac{1}{\tilde{n}})$.

Now, the idea is to not create the new population with p but with a distribution q that fulfills the equation

$$p(1-p) = xq(1-q) \tag{1.10}$$

with x fulfilling the equation:

$$1 - \frac{1}{\tilde{n}} = x(1 - \frac{1}{n}) \Leftrightarrow x = \frac{(\tilde{n}-1)n}{(n-1)\tilde{n}} \tag{1.11}$$

So we have:

$$p(1-p) = q(1-q)\frac{(\tilde{n}-1)n}{(n-1)\tilde{n}} \Leftrightarrow$$

$$-q^2 + q - (-p^2 + p)\frac{(n-1)\tilde{n}}{(\tilde{n}-1)n} = 0 \Leftrightarrow$$

$$q_{1/2} = \frac{1}{2}(1 \pm \sqrt{1 - 4(-p^2+p)\frac{(n-1)\tilde{n}}{(\tilde{n}-1)n}})$$

For the case of $\tilde{n} = \frac{n}{2}$ (i.e. we select half of the population to generate a new distribution vector) we would get

$$q_{1/2} = \frac{1}{2}(1 \pm \sqrt{1 - 4(-p^2 + p)\frac{(\tilde{n}-1)2\tilde{n}}{(2\tilde{n}-1)\tilde{n}}}) =$$

$$\frac{1}{2}(1 \pm \sqrt{1 - 4(-p^2 + p)\frac{2\tilde{n}-2}{2\tilde{n}-1}})$$

For $1 - 4(-p^2 + p)\frac{(n-1)\tilde{n}}{(\tilde{n}-1)n} < 0$ we get a negative value in square root. The border values for p are:

$$1 - 4(-p^2 + p)\frac{(n-1)\tilde{n}}{(\tilde{n}-1)n} = 0 \Leftrightarrow$$

$$p_{1/2} = \frac{1}{2}(1 \pm \sqrt{1 - \frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}})$$

So if our p is within p_1 and p_2

$$\frac{1}{2}(1 - \sqrt{1 - \frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}}) < p < \frac{1}{2}(1 + \sqrt{1 - \frac{(\tilde{n}-1)n}{(n-1)\tilde{n}}})$$

we cannot increase the variance any further and have to set $q = 0.5$.

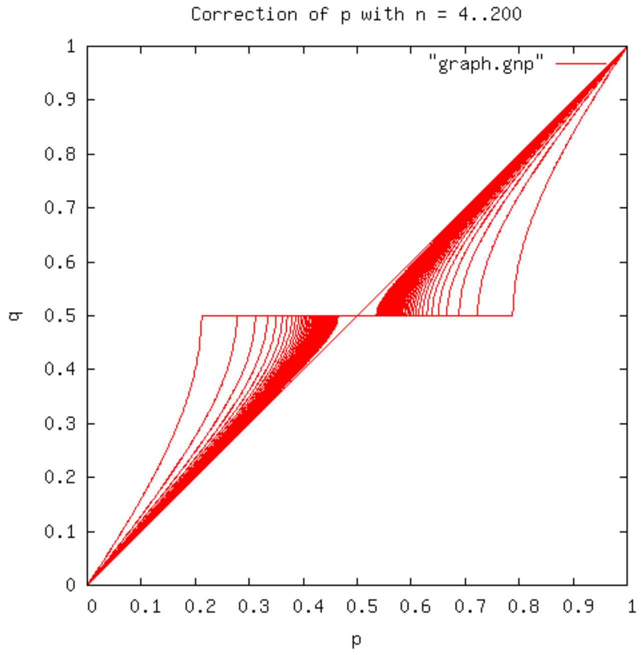


Figure 1.1: For a set of values for n (with $n = 2\tilde{n}$) ranging from 4 (farthest on the left and right) to 200 (nearest to $f(p) = p$)

1.4 Code

Finally we can put our results into a code ($N = \tilde{n}$):

```
if( p < 0.5*(1 - sqrt( (2*N-2) / (2*N-1) )) )
    q = 0.5 * (1 - sqrt( 4*p*(1-p) * (2*N-1) / (2*N-2) ) );
else
if( p > 0.5*(1 + sqrt( (2*N-2) / (2*N-1) )) )
    q = 0.5 * (1 + sqrt( 4*p*(1-p) * (2*N-1) / (2*N-2) ) );
else
    q = 0.5;
```

or for the general formula with $n \neq 2\tilde{n}$ ($M = n, N = \tilde{n}$):

```
if( p < 0.5 * (1 - sqrt( ((N-1)*M) / ((M-1)*N) ) ) )
    q = 0.5 * (1 - sqrt( 4*p*(1-p) * ((M-1)*N) / ((N-1)*M) ) );
else
if( p > 0.5 * (1 + sqrt( ((N-1)*M) / ((M-1)*N) ) ) )
    q = 0.5 * (1 + sqrt( 4*p*(1-p) * ((M-1)*N) / ((N-1)*M) ) );
else
    q = 0.5;
```

The main advantage of this code is that only minimal changes to the code have to be made because this way of reducing the diversity loss is problem independent (at least for bitstrings on a flat landscape in UDMA). The code has to be inserted just after determining the distribution p.

1.5 Multiple components

If we set $C > 1$, i.e. bitstrings longer than one bit, we will get similar results. While equation 3 does use C to determine the diversity of a generated population we can look at each component seperately as they are not connected in UMDA (in a flat fitness landscape). All bits belonging to one component in the population are created with an own distribution p that is independent from the other p. Therefor we can handle a problem with $C > 1$ like C seperate problems, each with one component. How interconnected components would affect the diversity is out of the scope of this paper and remains to be investigated.

1.6 Tests

1.6.1 General test configuration

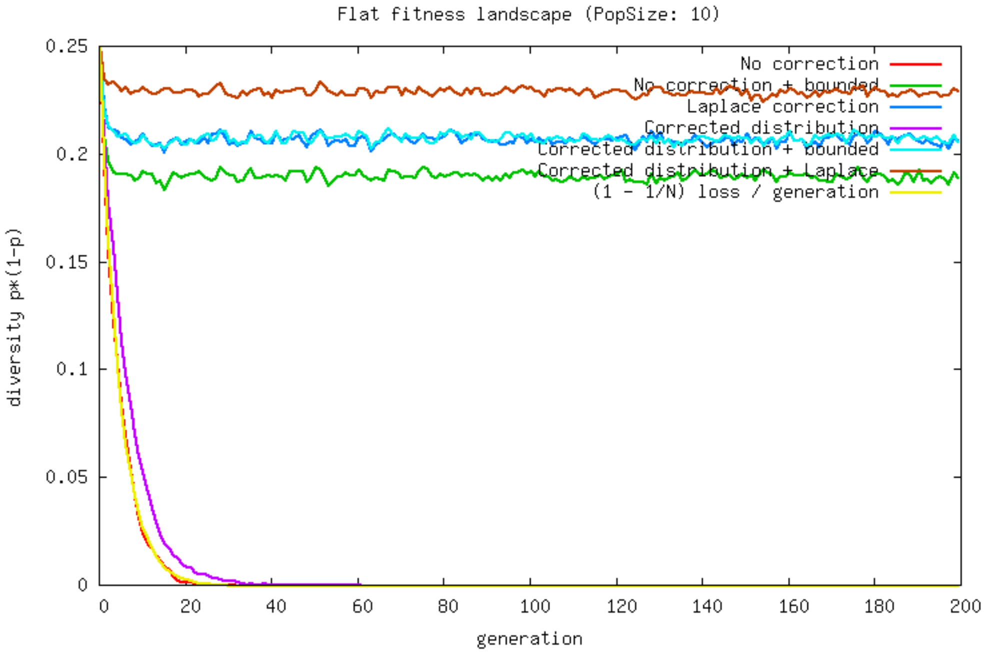
For both tests, OneMax and Flat fitness landscape, 50 seperate were repeatedly (with varying parameters) made.
6 different algorithms were tested:

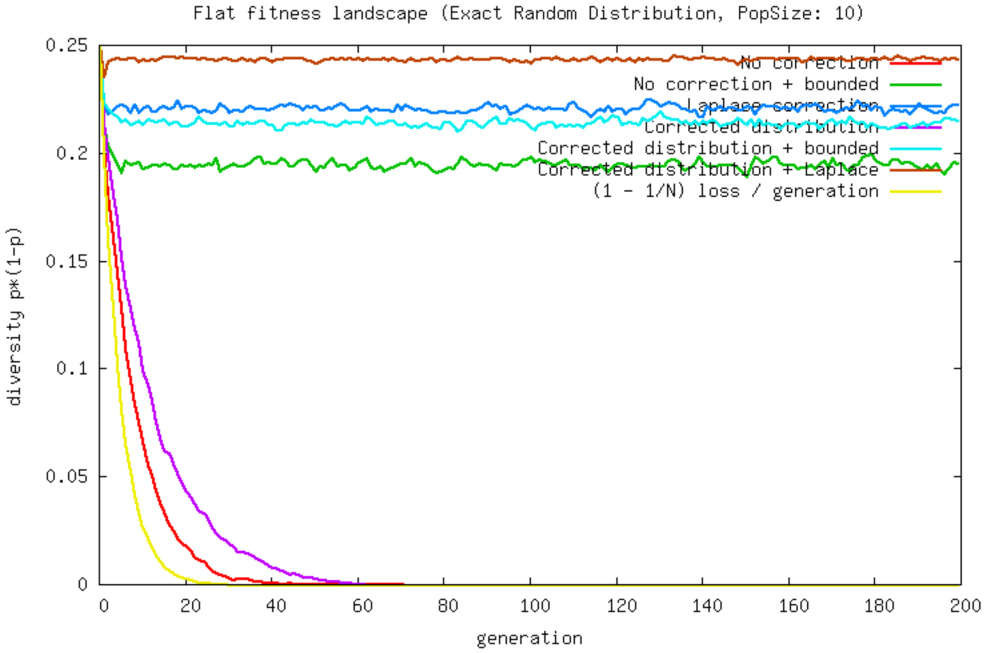
- **No correction**: The new distribution p for each component is simply calculated by dividing the number of 1's by the sample size
- **No correction + bounded**: Same as (1) but in the case p gets above $1 - \frac{1}{\tilde{n}}$ or below $\frac{1}{\tilde{n}}$ it is corrected to these boundaries
- **Laplace correction**: The new p is calculated by $\frac{k+1}{\tilde{n}+2}$
- **Corrected distribution**: Same as (1), but the resulting p is corrected with the formula discussed in the previous section
- **Corrected distribution + bounded**: Same as (2) but with the correction of p before checking the boundaries
- **Corrected distribution + Laplace**: Same as (3) but with the correction of p afterwards

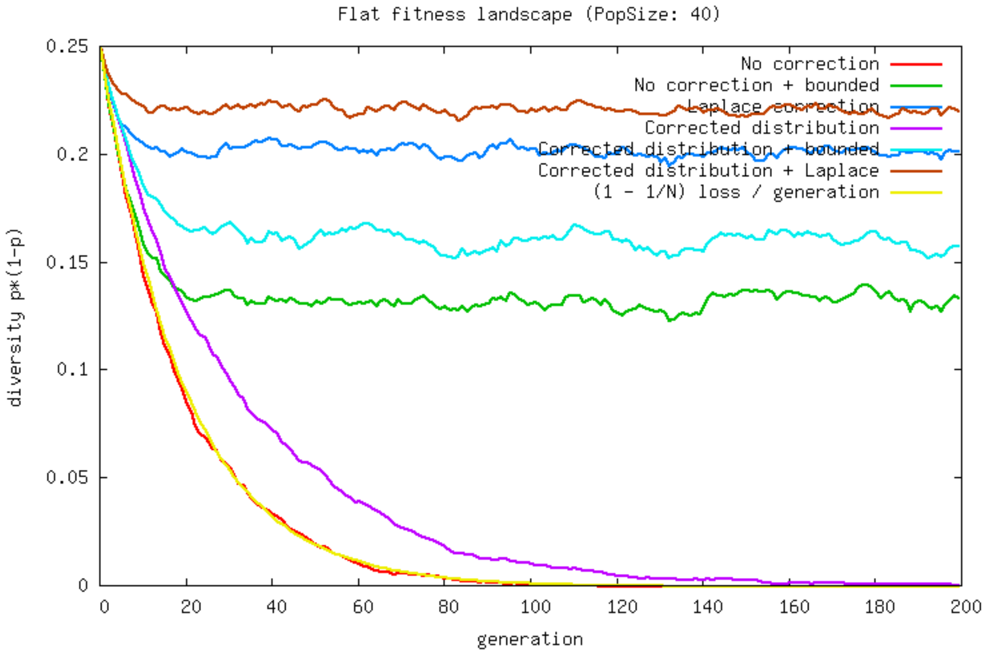
Using the for each component newly calculated distribution p we generate the new population of the size n depending on our parameter called 'Exact Random Distribution'. If it is not set we create each member of the population individually, if it is set we distribute pn '1' and $(1-p)n$ '0' between the individuals.
Tests have shown that the variance of the runs themselves are of no significance so it is are not included in the graphs.
The graphs were created with the help of gnuplot and C++, the actual program will be availble with a later version of this paper on CD-ROM.

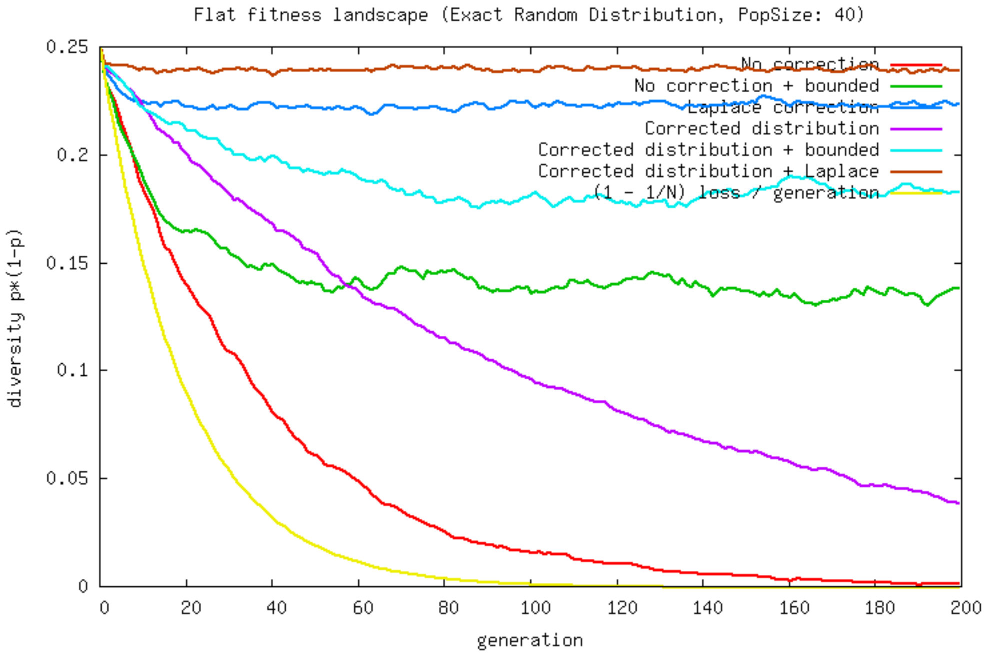
1.6.2 Flat fitness landscape

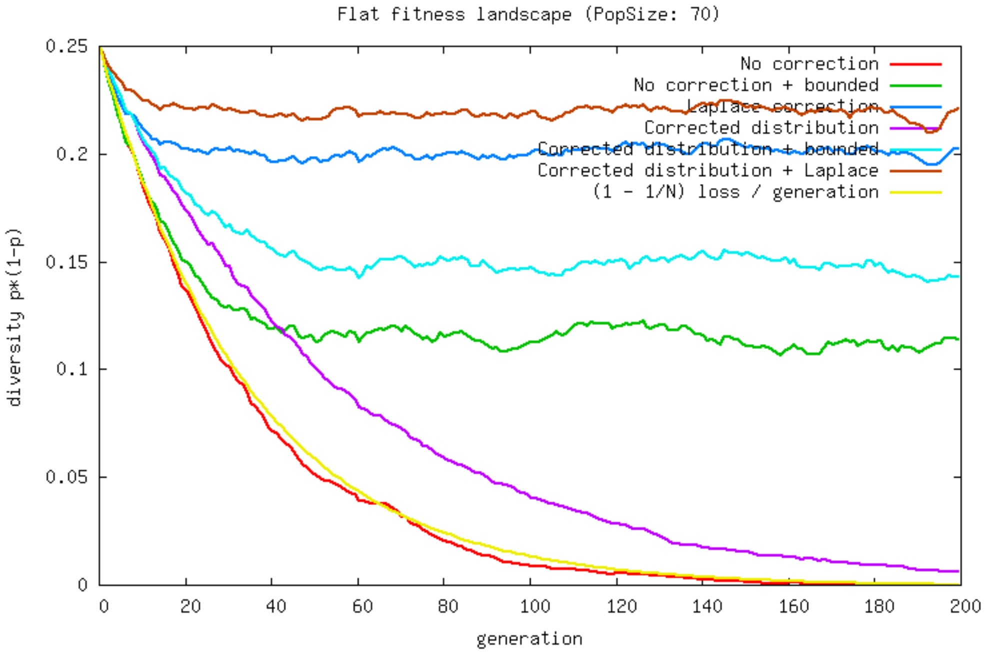
In this test we examine the behaviour of the algorithms in terms of their diversity with varying parameters on a flat fitness landscape. It is basicly a 'needle in a haystack' problem where the needle is not found within the 200 generations, i.e. all solutions have the same fitness. The problem size is a bitstring with length 10, i.e. we have 10 components.
The additional graph, '1 - 1/N loss / generation', denotes the theoretical loss of diversity according to [1].

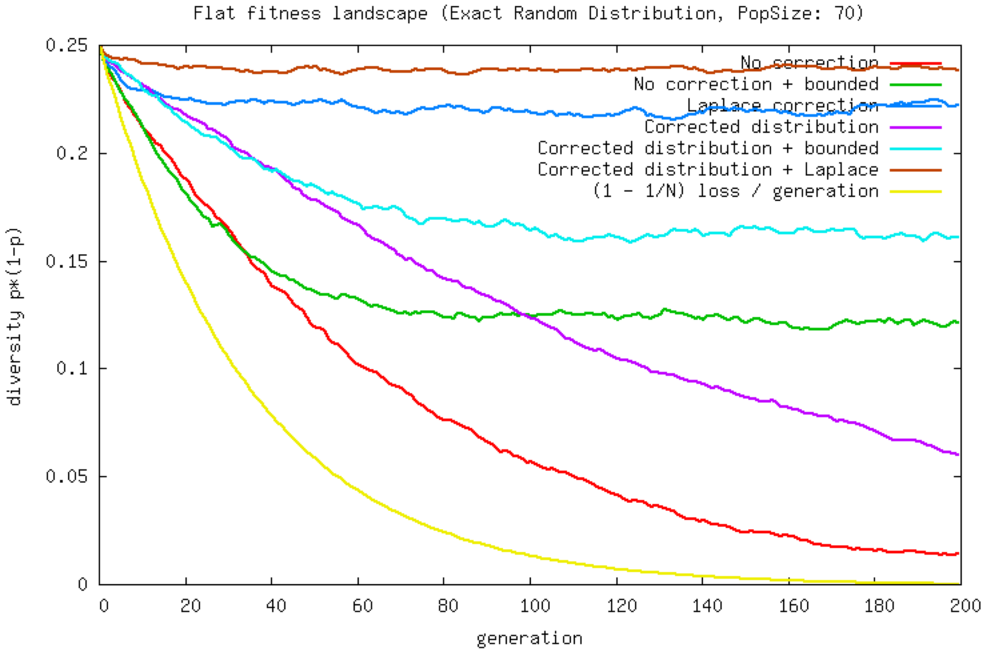

Flat fitness landscape (PopSize: 10)
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation
diversity p*(1-p)
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
120
140
160
180
200
generation

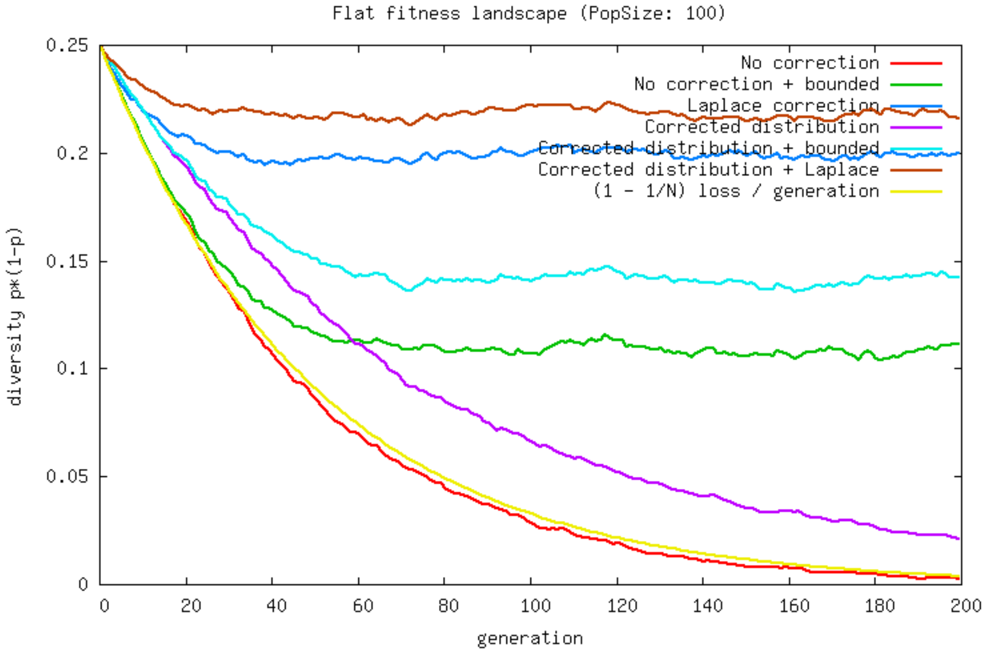

Flat fitness landscape (Exact Random Distribution, PopSize: 10)
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation
diversity p*(1-p)
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
120
140
160
180
200
generation

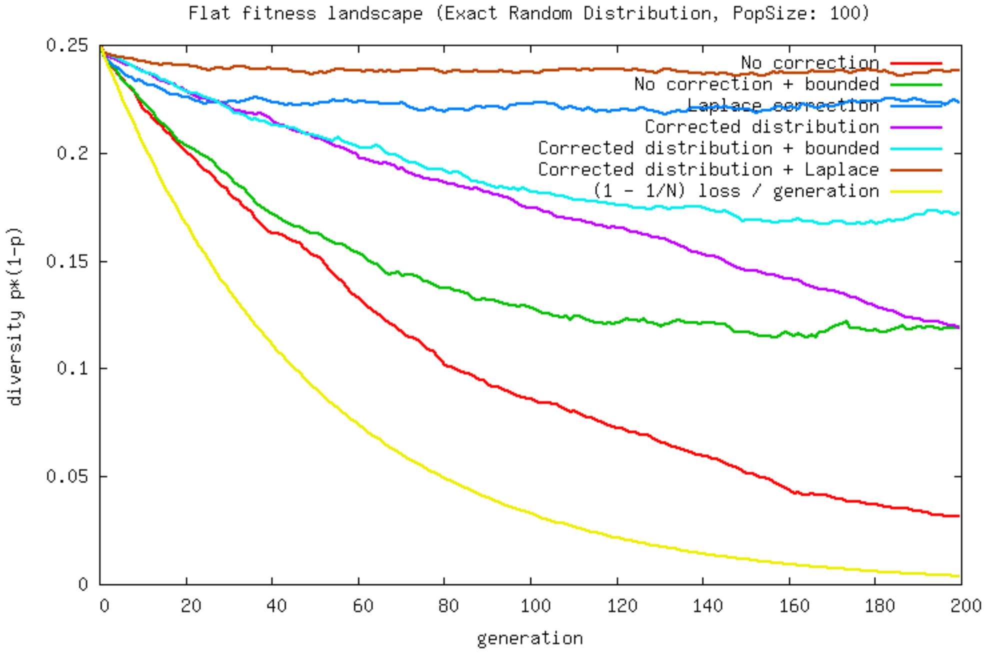

Flat fitness landscape (PopSize: 40)
diversity p*(1-p)
generation
0
0.05
0.1
0.15
0.2
0.25
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation


Flat fitness landscape (Exact Random Distribution, PopSize: 40)
diversity p*(1-p)
generation
0
0.05
0.1
0.15
0.2
0.25
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation


Flat fitness landscape (PopSize: 70)
diversity p*(1-p)
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
120
140
160
180
200
generation
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation

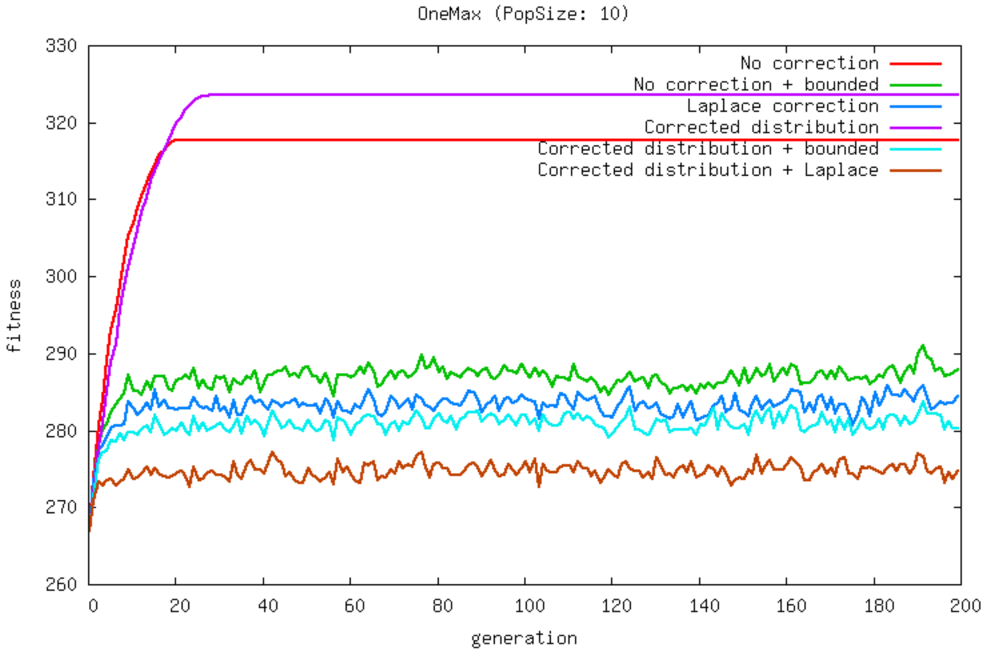

Flat fitness landscape (Exact Random Distribution, PopSize: 70)
diversity p*(1-p)
0.25
0.2
0.15
0.1
0.05
0
0
20
40
60
80
100
120
140
160
180
200
generation
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation

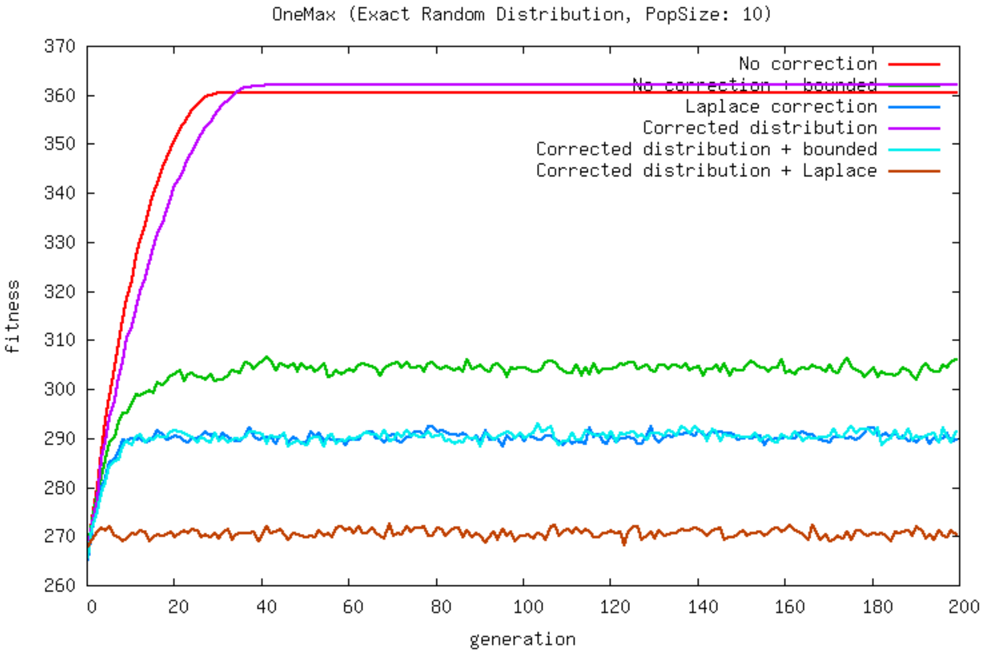

Flat fitness landscape (PopSize: 100)
diversity p*(1-p)
0.25
0.2
0.15
0.1
0.05
0
generation
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation

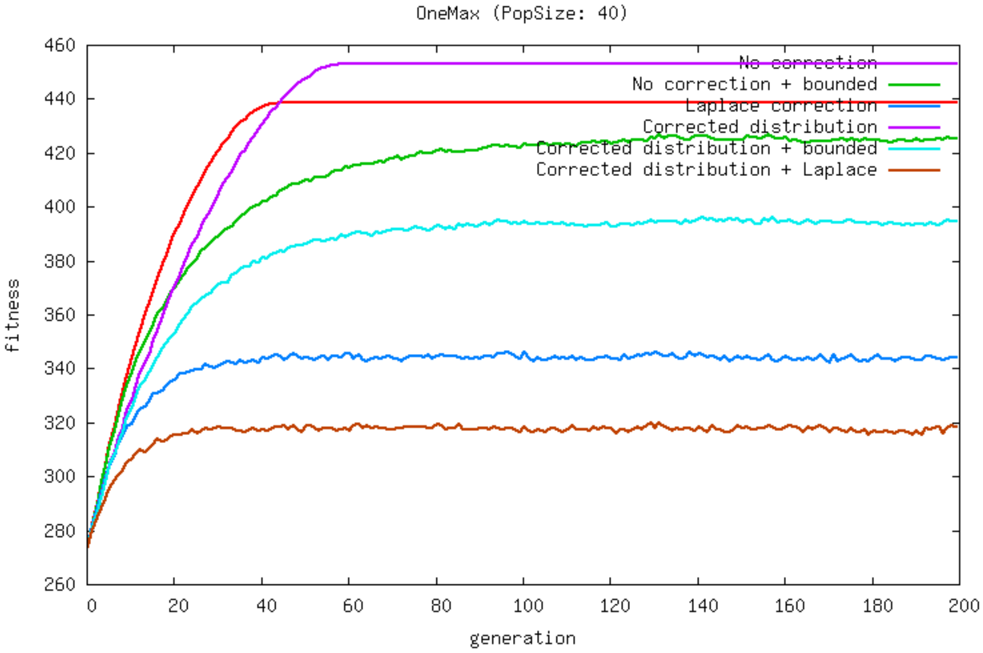

Flat fitness landscape (Exact Random Distribution, PopSize: 100)
diversity p*(1-p)
0.25
0.2
0.15
0.1
0.05
0
generation
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace
(1 - 1/N) loss / generation

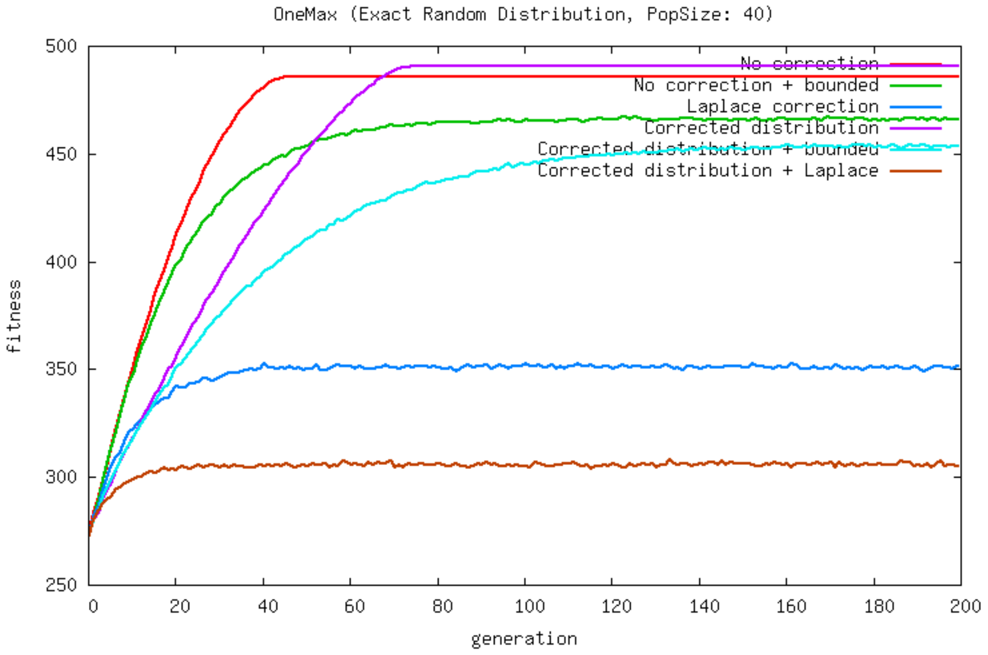
1.6.3 OneMax problem

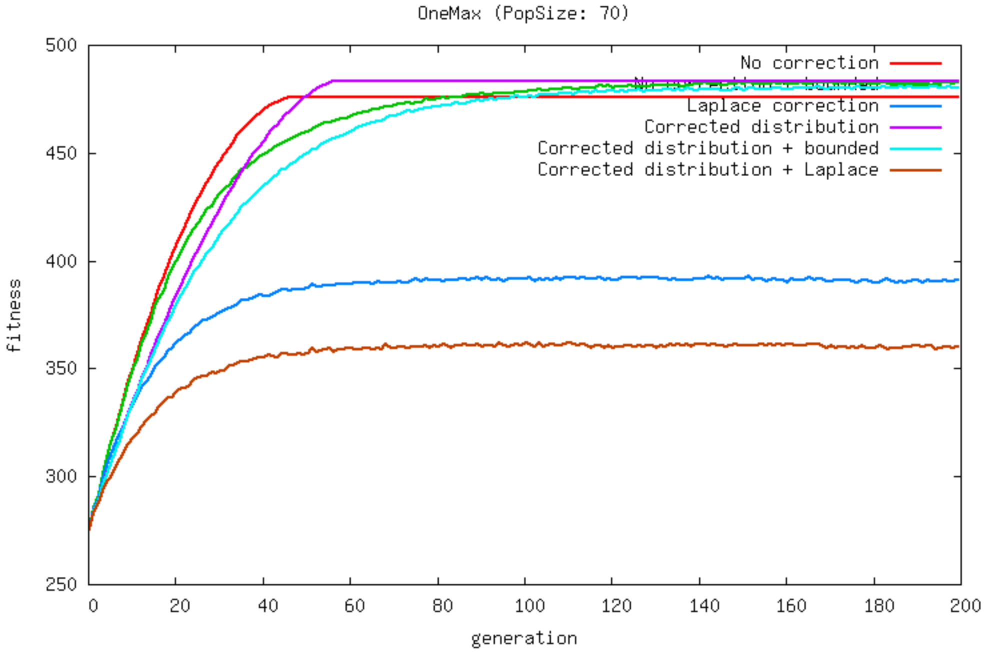
In this test we examine the behaviour of the algorithms in terms of their fitness with varying parameters with the problem OneMax. The goal is to find the string '111...', each '1' in a component of an individual gets rewarded by one fitness point. In our test the problem size if 500.
For each generation only the highest fitness of each population is taken into account. Again the graph shows the average of the fitness values of all runs.

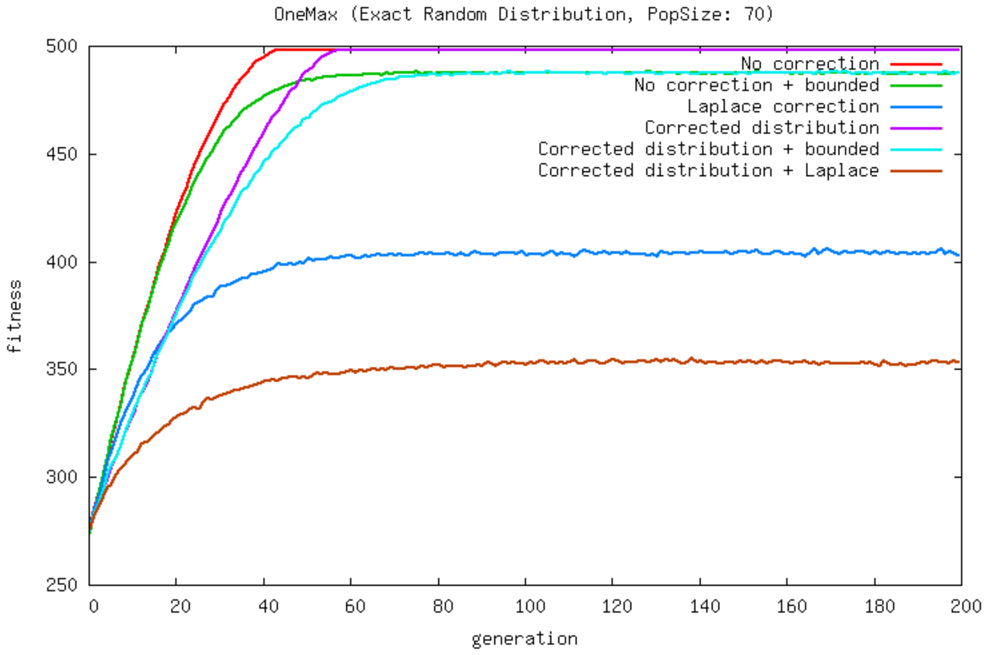


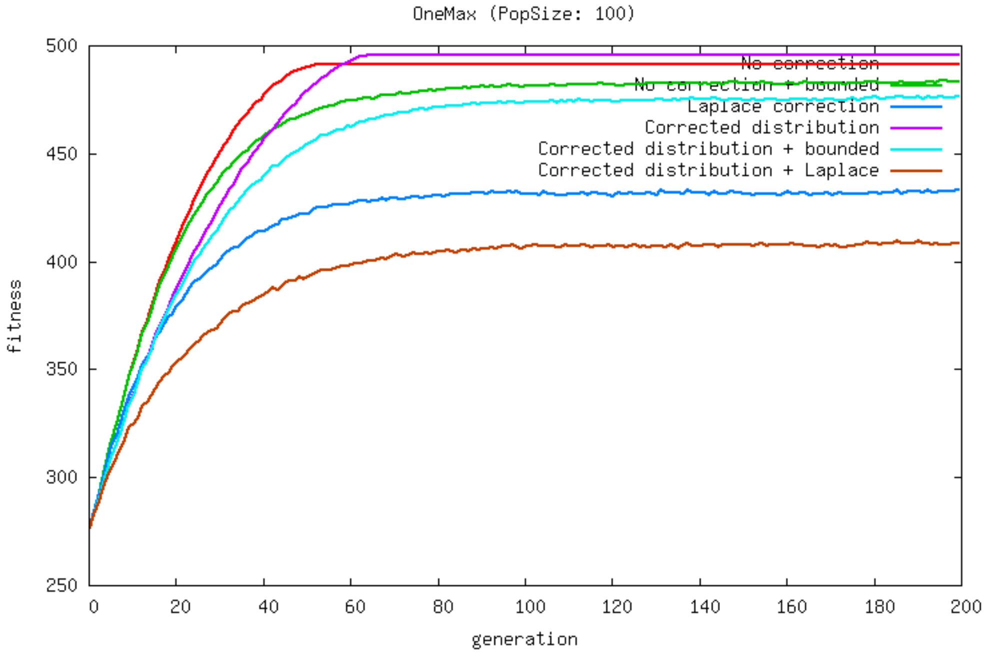

OneMax (Exact Random Distribution, PopSize: 10)
fitness
generation
370
360
350
340
330
320
310
300
290
280
270
260
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace

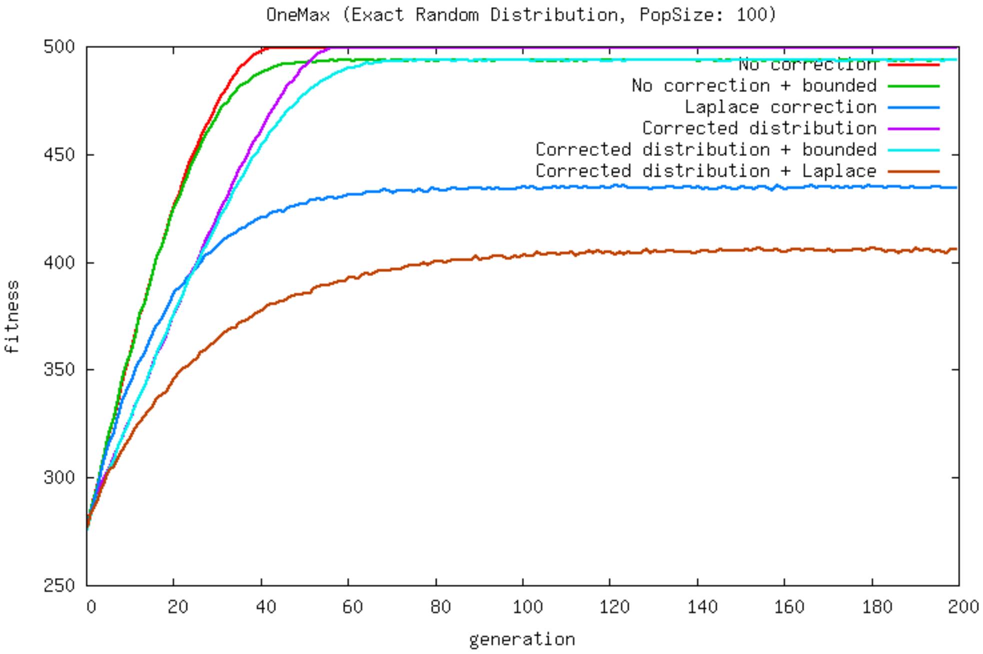

OneMax (PopSize: 40)
fitness
generation
460
440
420
400
380
360
340
320
300
280
260
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace


OneMax (Exact Random Distribution, PopSize: 40)
fitness
generation
500
450
400
350
300
250
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace


OneMax (PopSize: 70)
fitness
generation
500
450
400
350
300
250
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace


OneMax (Exact Random Distribution, PopSize: 70)
fitness
generation
250
300
350
400
450
500
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace


OneMax (PopSize: 100)
fitness
generation
250
300
350
400
450
500
0
20
40
60
80
100
120
140
160
180
200
No correction
No correction + bounded
Laplace correction
Corrected distribution
Corrected distribution + bounded
Corrected distribution + Laplace



1.7 Conclusions

1.7.1 Flat fitness landscape

With the tests we have shown that our 'Corrected distribution' algorithm does significantly reduce the diversity loss compared to an algorithm with no correction. Making sure that the algorithm does not stall in a component with $p < \frac{1}{n}$ or $p > 1 - \frac{1}{n}$ ("bounded") makes sure that the diversity does not drop to zero in the long run. 'Corrected distribution' does not outperform 'Laplace correction' as the latter always corrects any distribution towards $p = 0.5$.

1.7.2 OneMax

While the OneMax problem does not represent a problem with a real flat fitness landscape (thus our theoretical base does not apply here), a flat fitness landscape can occur if the variance drops and/or the fitness values are very similar (e.g. 010, 100, 001).
'Corrected distribution + Laplace' is clearly worse than 'Laplace correction', the changes made to the distribution p (that we determined on base of the selected individuals) are too big in order to get useful results. As expected 'Laplace correction' itself does not score well, it converges at a significant lower fitness level.
'Exact random distribution' seems to generally improve the convergence of methods using not the 'Laplace correction' while being indifferent or being even worse for the 'Laplace correction' itself.

At least for the case of 'OneMax' and the given parameter configuration, 'Corrected distribution' in connection with 'Exact Random Distribution' seems to outperform all other methods.

1.8 Fields of further research

According to [1] the diversity loss of $1 - \frac{1}{n}$ is independent of $|A|$ and is valid for a whole class of so-called SML-EDAs, including UMDA, MIMIC, FDA or BOA. In this paper I have assumed that $|A| = 2$, i.e. that a component can only take two different values. With values $|A| > 2$ the proof has to be adapted from equation (5) on, depending on how one creates such a population. It also remains to be investigated how a similar correction of the diversity loss is possible with other methods than UMDA that fall into the category of the SML-EDAs. It is probable that the idea itself, i.e. determining the loss of variance due to sampling and calculating it backwards in order to correct the distribution p, seems to be applicable to many different forms of SML-EDAs or even EDAs in general - assuming one can calculate the actual diversity loss.

Bibliography

[1] SHAPIRO, J.L.: *Diversity loss in general estimation of distribution algorithms*, 2006.