An Evolution Strategy using a continuous version of the Gray-code neighbourhood distribution

Jonathan E. Rowe and Džena Hidović

School of Computer Science, University of Birmingham, Birmingham B15 2TT, Great Britain
J.E.Rowe@cs.bham.ac.uk, D.Hidovic@cs.bham.ac.uk

Abstract. We derive a continuous probability distribution which generates neighbours of a point in an interval in a similar way to the bitwise mutation of a Gray code binary string. This distribution has some interesting scale-free properties which are analogues of properties of the Gray code neighbourhood structure. A simple (1+1)-ES using the new distribution is proposed and evaluated on a set of benchmark problems, on which it performs remarkably well. The critical parameter is the *precision* of the distribution, which corresponds to the string length in the discrete case. The algorithm is also tested on a difficult real-world problem from medical imaging, on which it also performs well. Some observations concerning the scale-free properties of the distribution are made, although further analysis is required to understand why this simple algorithm works so well.

1 Introduction

There are two different approaches to solving continuous-value optimisation problems using Evolutionary Computation. The first is to represent points in the search space using real numbers and to generate new points using some continuous probability distribution (typically Gaussian or Cauchy). The second approach is to discretise the space and represent real numbers as binary strings. One then mutates the strings by flipping one or more bits. It is known that there can be problems with this second approach if the standard binary encoding is used: there exist so-called Hamming cliffs — points that are neighbours according to the topology of the space, but are not neighbours when considered as binary strings. An alternative representation is to use a Gray code, in which all neighbours in the original space are also neighbours as strings. The trade-offs between using the standard binary and Gray representations have been studied in some detail by Whitley [1]. It can be shown that on some classes of optimisation problem, the Gray code representation has definite advantages. For example, it can be shown that a local search algorithm using this representation can solve a one-dimensional unimodal problem in quadratic time, and a clever variant can do it in linear time [2].

As part of the theoretical investigation of the use of Gray codes, one can ask about the distribution of neighbours that a point has, under this encoding. Suppose we use ℓ bits to represent the numbers $0, 1, \ldots, 2^\ell - 1$. Given a point x in this range, we want to know something about the set of its neighbours, generated by flipping exactly one bit of the Gray code representation of x. For example, if $\ell = 4$ and $x = 13 = 1011$ in Gray code, then the neighbours of x are $2 = 0011, 10 = 1111, 12 = 1010, 14 = 1001$. One way to charactise this question in general terms is to ask: given a point x, how many neighbours of x are within a given distance t? It can be shown that, on average, there will be $\lfloor \lg t + 1 \rfloor$ such neighbours [3].

It would be nice, from a theoretical point of view, if we could relate this method of local search to the standard Evolution Strategies (ES) which make use of real-valued representations and generate neighbours using continuous probability distributions. In other words, we ask the question: is there a continuous probability distribution such that the probability of generating a neighbour within a given interval is the same as if we used a Gray code representation and bitwise (point) mutation? This is the question we address, and answer, in section 2. We would also like to know what properties an Evolution Strategy using this new distribution has. In particular, are there theorems analogous to those already proved for the discrete Gray code local search algorithm? We investigate some of these properties in section 3.

Having developed this distribution and analysed the corresponding ES from a theoretical point of view, an obvious question arises: is it any good for optimisation? We study its performance on a collection of standard benchmark problems in section 4, comparing its performance (under various settings of the main contol parameter) with a recently published Evolutionary Programming algorithm (the "Improved Fast Evolutionary Programming" algorithm [4]) which makes use of a population and self-adaptive mutation rates and mixed Gaussian-Cauchy mutation distributions (see also [5]). The conclusion is that, remarkably, the simple (1+1)-ES with the new distribution is exceptionally good.

Of course, benchmark problems are one thing, and real-world applications another. So we conclude by presenting some results from a difficult problem in medical tissue optics: finding the values of structural parameters describing colon tissue that could give rise to observed colours in colonoscopy images. This is an important application area in medicine: the ability to distinguish normal from cancerous colon tissue optically would reduce the need for biopsies and assist clinicians in making diagnoses [6]. Again, the new algorithm is compared to the IFEP algorithm, and performs remarkably well (section 5).

2 The continuous version of the Gray neighbourhood distribution

In this section we will derive a continuous probability distribution which has properties directly analagous to the discrete Gray code representation under the usual definition of Hamming neighbourhoods. The key properties which we emulate are:

– mutations of a bit string generate moves with a *minimum* distance, specified by the precision of the code (or equivalently, the string length).
– the probability of producing a neighbour within a distance of t discrete points of the current point is, on average, $\lfloor \lg t + 1 \rfloor / \ell$.
– the Gray code naturally represents a bounded interval.
– the Gray code "wraps around": the strings corresponding to $2^{\ell} - 1$ and 0 are Hamming neighbours. The *maximum* distance of a move is thus half the search space (in either direction).

To keep things simple, we will assume that we have a one-dimensional search space which is the interval $[-1, +1]$. Any other bounded interval can be mapped by an affine transformation into this standard interval.

$$\varphi : [a, b] \longrightarrow [-1, +1]$$

$$\varphi(x) = 2\left(\frac{x-a}{b-a}\right) - 1$$

We need to specify a *minimum step size* which we denote ε. Equivalently, we will define the *precision* to be $p = -\log\varepsilon$. The precision is an analogue of the string length of the Gray code. The maximum step size will be half the size of the interval (in either direction): that is, 1. We define a probability density function

$$f(x) = \begin{cases} \frac{1}{px} & \text{if } \varepsilon < x < 1 \\ \\ 0 & \text{otherwise} \end{cases}$$

We will choose the distance between the current point and the new point (the neighbour) according to this density function, moving to the left or right with equal probability. Thus the probability of picking a neighbour within a distance τ of the current point is

$$\int_{\varepsilon}^{\tau} \frac{dx}{px} = \frac{\log\tau}{p} + 1$$

By analogy to the discrete Gray encoding, let t be the number of minimal steps needed to move a distance of τ. That is $\tau = \varepsilon t$. Then the probability of jumping within a distance τ becomes

$$\frac{\log\varepsilon t}{p} + 1 = \frac{\log t}{p}$$

This defines, therefore, a continuous probability distribution which distributes neighbours in a way exactly analagous to the Gray code representation. But what should we do if the distance to be moved takes us outside the range $[-1, +1]$? We will simply wrap around in the same way that the Gray code does.

All that remains to be able to write an Evolution Strategy based on this distribution is a method for generating random numbers according to the given distribution. To do this, we note that the cumulative distribution function is:

$$F(x) = \int_{-\infty}^{x} f(t)dt = 1 + \frac{\log x}{p}$$

We can generate a random number according to this distribution by first generating a random number u uniformly from $[0, 1]$ and then setting $\tau = F^{-1}(u) = \exp(-p(1-u))$. Equivalently, we can set $\tau = \exp(-pu)$ since $1-u$ is also distributed uniformly in $[0, 1]$. See [7] for more details of this method.

Suppose $g : [-1, 1] \to \mathbb{R}$ is the objective function, and, without loss of generality, that we are minimising. We define our (1+1)-ES as follows:

1. Pick an initial point $x \in [-1, +1]$.

2. Generate a random number u uniformly in $[0, 1]$.
3. Set $\tau = \exp(-pu)$.
4. With probability half, set $y = x + \tau$, else $y = x - \tau$.
5. If $y < -1$ set $y = y + 2$. If $y > 1$ set $y = y - 2$.
6. If $g(y) < g(x)$, set $x = y$.
7. Go to 2.

3 Properties of the distribution

We have shown that our new continuous probability distribution generates neighbours of a point that are distributed analagously to the discrete Gray code representation. We now look at some other properties that the Gray code has, and derive corresponding results for our new distribution.

The Gray code neighbourhood structure has some remarkable *scale invariant* properties. Firstly, it is clear from the recursive construction of the Gray code that any point has neighbours at all scales. That is, if the point is in one half of the search interval, it has a neighbour in the other half. Zooming in, if we consider the quarter of the search interval containing the point, then there is a neighbour in an adjacent quarter. One can continue to zoom in, throwing away half the interval at each step, and one will always find neighbours. When it comes to the continuous distribution, there is, of course, a non-zero probability of generating a neighbour right across the search interval. However, by a “scale free” distribution is meant one in which the probability of finding points at any distance is not vanishingly small. So the Gaussian distribution, for example, while assigning a non-zero probability across the range, has tails that shrink exponentially: it is therefore not scale-free. With our new distribution, however, one never has to wait too long for jumps of arbitrarily large size (up to the maximum). The maximum jump size is 1. The probability of making a jump bigger than $1 - \delta$ is

$$\int_{1-\delta}^{1} \frac{dx}{px} = -\frac{\log(1-\delta)}{p}$$

so the expected waiting time is $O(p)$.

The second scale-invariant property shows up in the analysis of the steepest descent Gray code algorithm applied to a one-dimensional unimodal function, in which it takes a constant number of trials in order to disregard half of the remaining search interval under consideration. We have a similar result here. Suppose the current point is a distance z away from the optimum. The probability of making one jump that would take us within $z/2$ of the optimum is

$$\frac{1}{2}\int_{z/2}^{z} \frac{dx}{px} = \frac{\log 2}{2p}$$

That is, it is independent of the current position! The expected waiting time (and this is clearly an upper bound) is thus $2p/\log 2$. The number of steps required to get within δ of the optimum is therefore $O(p\log(1/\delta))$. This result might make one think that it is best to choose the precision p to be as small as possible, but of course, one needs a sufficiently small minimum step size to be able to approach the optimum as closely as desired.

4 Experiments with the ES

Having developed a simple search algorithm for theoretical purposes, it seemed worth trying it out on a range of test problems. Partly this was to investigate the effects of varying the precision parameter on the performance of the algorithm, but we also wished to see if its performance were comparable with other evolutionary optimisation algorithms. Consequently, we took eight benchmark problems from the paper [4] which introduced a new Evolutionary Programming algorithm called Improved Fast Evolutionary Programming (IFEP). We used a variant of the IFEP algorithm to provide a baseline performance against which we compared our algorithm. Specifically, we used a (15,45)-ES in which each population member produces three offspring. A single offspring is produced by mutating according to both Gaussian and Cauchy distributions and taking the best (thus each offspring requires two fitness evaluations). The mutations are self-adaptive, as described in the paper. The best 15 individuals are chosen from the offspring to form the next population. [1]

The test functions are taken from the same paper and are defined as follows (note that the minimum is zero in each case):

Sphere function
$f_1(x) = \sum_{i=1}^{30} (x_i)^2 \qquad x_i \in [-100, 100]$

Schwefel's problem 2.22
$f_2(x) = \sum_{i=1}^{30} |x_i| + \prod_{i}^{30} |x_i| \qquad x_i \in [-10, 10]$

Schwefel's problem 1.2
$f_3(x) = \sum_{i=1}^{30} \left(\sum_{j=1}^{j=i} x_j \right)^2 \qquad x_i \in [-100, 100]$

Schwefel's function 2.21
$f_4(x) = \max\{|x_i|, 1 \le x_i \le 30\} \qquad x_i \in [-100, 100]$

Generalised Rosenbrock's function
$f_5(x) = \sum_{i=1}^{29} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \qquad x_i \in [-30, 30]$

Generalised Rastrigin's function
$f_9(x) = \sum_{i=1}^{30} (x_i^2 - 10\cos(2\pi x_i) + 10) \qquad x_i \in [-5.12, 5.12]$

Ackley's function
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2}\right) - \exp\left(\sum_{i=1}^{30} \cos 2\pi x_i\right)$
$+20 + e \qquad x_i \in [-32, 32]$

Generalised Griewangk function
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \qquad x_i \in [-600, 600]$

[1] Although this is a variant on the IFEP algorithm, the results we obtained are largely similar to those reported for IFEP.

For multi-dimensional problems such as these, we have to adapt our Evolution Strategy slightly. At each iteration, we mutate *all* of the parameters of the current point simultaneously, using the method described in the previous section.

We conducted a number of experiments with these test functions, varying the precision parameter p through the values 25, 50, 100 and 200. We also considered the effect of the number of iterations allowed (1000, 10000, 100000 and 1000000). We adjusted the number of generations allowed to the IFEP algorithm accordingly, to get the same total number of function evaluations. Each experiment was run 30 times. The average results (function values) are shown in figure 1. Standard deviations are not shown, but nearly all differences are significant at the 99.9% level according to a two-tailed t-test. Note that each graph is shown on a log-log scale.

It is clear that our new (1+1)-ES has performed very well, especially over large iterations with a high precision value. What is also clear from the data is that it is often significantly better than the IFEP algorithm over a small number of iterations, when a lower precision value is used. It also gives results that are comparable to the best results reported for various evolutionary algorithms in [5] [2]. We also ran some experiments with a (1+1)-ES with Gaussian mutation using the $1/5$ success rule. As might be expected, this algorithm performs extremely well on the sphere function. It performs moderately well on other unimodal functions. However it is terrible on multimodal functions — by construction, it is designed to converge rapidly to the nearest local optimum. The new algorithm (and indeed IFEP) is superior on such functions.

5 A real-world application

Having tested our algorithm on some standard benchmark problems, we then applied it to a difficult real-world problem, from medical image interpretation. An increasingly important application of image interpretation is the development of non-invasive techniques for studying tissue structure. Clinicians want to be able to deduce as much as possible about the structure of an organ from its visual images, to reduce the necessity of performing biopsies. One approach to this problem focuses on analysing the physics of image formation. The basic idea is to create a physics-based model of the tissue structure and to simulate the effect of shining white light onto the surface. As a result of that simulation, the amount of light that reemerges at the tissue surface (spectral reflectance), after interacting with the tissue structure, is calculated. By adjusting the parameters of the model, one tries to reproduce the optical spectra measured on real tissue, in order to analise the corectness of the model. One can then, in principle, match the spectra with the appropriate physical parameters and extract diagnostically valuable information about the tissue structure. However, it is rather difficult to establish all the relevant parameters and the corresponding value ranges for them. The initial stages of this research depend, therefore, on using optimisation algorithms to try to establish suitable parameter settings. For a more detailed description of the problem, see [6].

[2] Exact data are not presented in that paper. However, from the graphs shown it is clear that our new algorithm is comparable or better on all test functions than the algorithms presented in that paper.

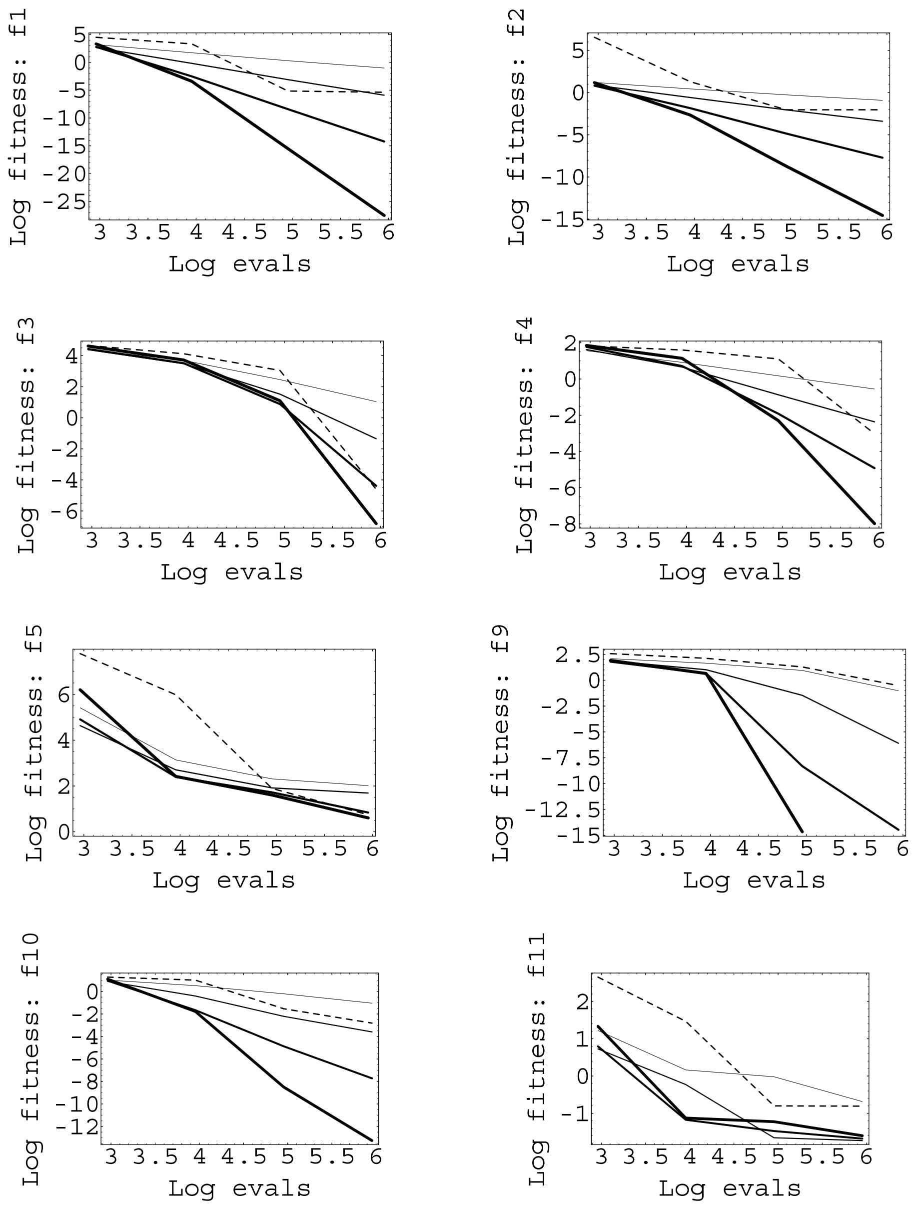

Fig. 1. Comparison of performance of the new (1+1)-ES (solid lines) and IFEP (dashed line). There are four different settings of the precision parameter ($p = 25, 50, 100, 200$), indicated by increasing thickness of the line used. Performance tends to improve with increasing precision.

We have obtained a set of spectra from normal colon tissue taken during colonoscopy procedures, where an optical fibre bundle (which delivers and collects light) is passed through a working channel of a colonoscope, and placed against the colon wall of a patient [8]. In each case an observed spectrum is collected. We then use our optimisation algorithm to try to find parameter settings in the physics-based model which will account for the observed spectrum.

The physics-based model of colon tissue is developed so that it simulates the interaction of incident light with the structure and morphology of the real colon tissue, which is a layered structure composed of four layers (mucosa, submucosa, muscularis externa and serosa). Our model predicts the light interaction with the first three layers, because the spectral reflectance of the colon tissue depends on the interaction of the light with only those layers.

The parameters of the model which describe the optical properties of the colon layers and hence directly influence the remitted spectrum, are:

– *haemoglobin concentration* is the amount of hemoglobin per unit volume of tissue. It describes the absorption of light in the colon tissue.
– *scatterer size* represents the size of collagen fibres in colon tissue, given that the collagen is the main scatterer of light in colon.
– *scatterer density* is the number of scattering particles (collagen fibres) per unit volume of the tissue.
– *thickness* of each of the tissue layers included in the model

These four parameters must be specified for both the mucosa and submucosa separately. The third layer (muscularis externa) is represented by a fixed set of values. In addition there is a scaling factor to account for adjustments to the normalisation process, in which the amount of collected light is divided by the amount of light reflected from a reflectance standard. There are therefore nine parameters to be optimised.

We use the Kubelka-Munk algorithm, [9, 10], to calculate the spectrum corresponding to a given set of parameter values. This is an approximate algorithm for calculating the diffuse reflectance of light from a layered structure. Greater accuracy could be obtained using the Monte Carlo method [11], but that takes much longer to execute (several minutes per run).

We seek to minimise the error between the generated spectrum and the target (measured) spectrum at 113 wavelengths equally spaced in the range from 400 nm to 624 nm. The error is calculated as average absolute distance between the corresponding spectral values:

$$d(y,z) = \frac{1}{n}\sum_{i=0}^{n} |y_i - z_i|$$

where y_i and z_i are the values of measured and simulated spectra corresponding to the wavelength w_i, and n is the total number of wavelengths. Due to the time it takes to run the Kubelka-Munk algorithm (approximately one second per run), we allow only 1000 function evaluations. The precision is set to 20. We again compare with the IFEP algorithm, with the number of generations adjusted to give the same number of function evaluations. The results are shown in table 1. The new Evolution Strategy is clearly superior (significance $> 99.99\%$ on a paired t-test). Some typical results are shown in figure 2.

Table 1. Performance of new (1+1)-ES and IFEP finding parameters for a physics-based model of colon colouration. Number of samples = 45.

Algorithm	Mean Error	Standard Deviation
(1+1)-ES	0.0178	0.0081
IFEP	0.0391	0.0121

6 Discussion and Conclusions

We have introduced a new Evolution Strategy, with a mutation probability distribution based on a continuous version of the Gray code neighbourhood structure. The distribution we have defined has certain *scale-free* properties which may be assisting its performance in search. In particular, while the algorithm spends a lot of its time searching locally, it nevertheless samples at a longer range with non-trivial probability (see figure 3). The mean of the distribution distribution is

$$\int_{\varepsilon}^{1} \frac{dx}{p} = \frac{1-\varepsilon}{p} \approx \frac{1}{p}$$

The probability of choosing a distance larger than the mean is approximately $\log p/p$. So, for example, with a precision of $p = 100$, the algorithm spends 95.4% of its time within 0.01 of its current position, but searches outside of this area with probability 0.046, which means that a “long-distance” jump (that is, one greater than the mean) occurs on average every 21.7 iterations. We also note that the standard deviation of the distribution is approximately $1/\sqrt{2p}$ which is rather large. For example, with $p = 100$, the mean is 0.01 and the standard deviation is 0.07.

However, we are still some way from understanding the effects of changing the precision on the performance of the algorithm, apart from the simple one-dimensional unimodal case. Further experiment and analysis are required. From a theoretical point of view, it is interesting to refer this question back to the case of a local search algorthim using binary strings under Gray code: what is the effect of changing the precision (that is, the string length) in this case?

The search algorithm seems to be rather useful, especially when a relatively small number of function evaluations are allowed. This is often the case in real-world applications, such as the one presented above, when fitness can be very expensive to calculate. It seems that having a relatively low precision works well over a short number of iterations, although this is yet to be demonstrated theoretically. Of course it is possible that an adaptive scheme, with increasing precision over time, may be worth investigating.

One nice feature of the new algorithm is that it applies naturally to bounded optimisation problems, which are a very common class. The distance distribution, together

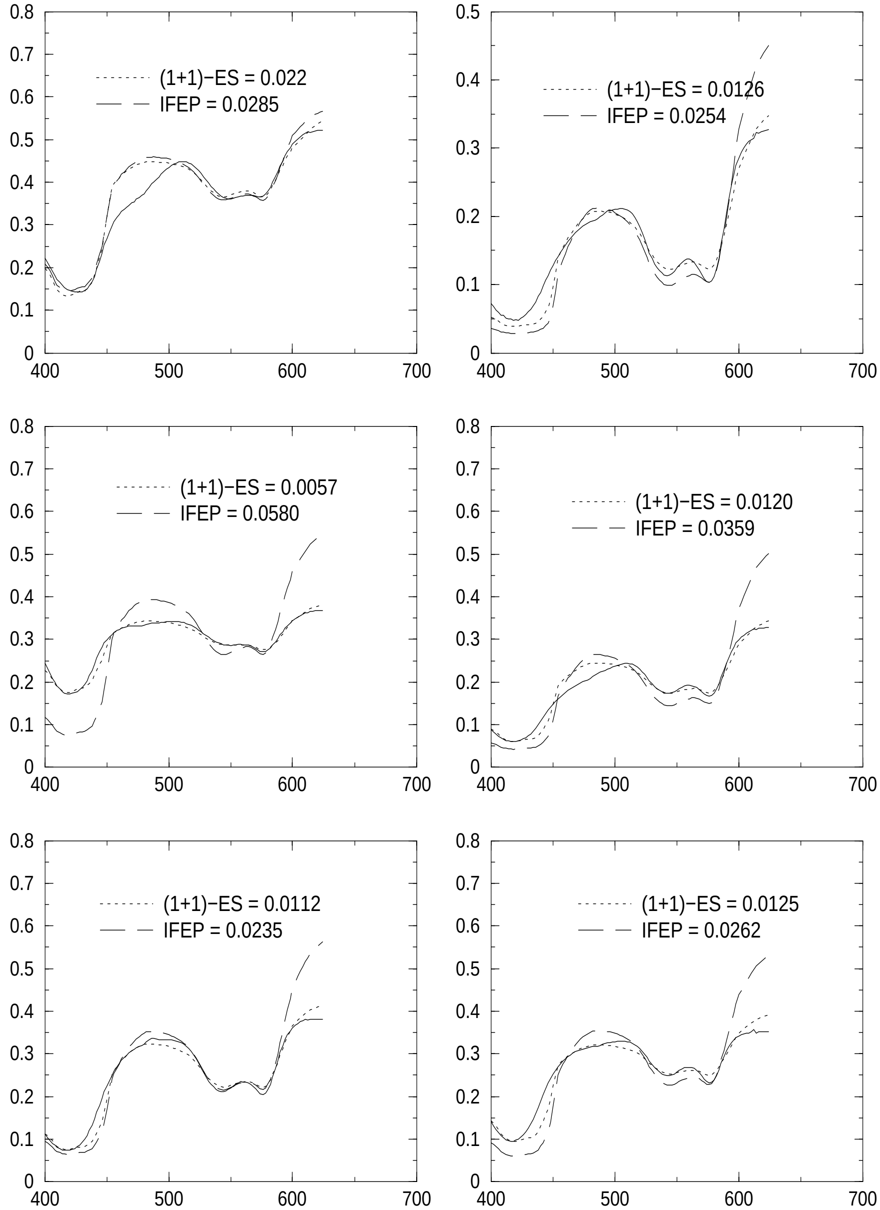

Fig. 2. Six example spectra from normal tissue(solid lines) obtained during colonoscopy. The new (1+1)-ES (dotted lines) and IFEP (dashed lines) algorithms were both used to find parameter settings to approximate these spectra, using the Kubelka-Munk method. Errors are measured as the average absolute distance from the generated curve and the target spectrum.

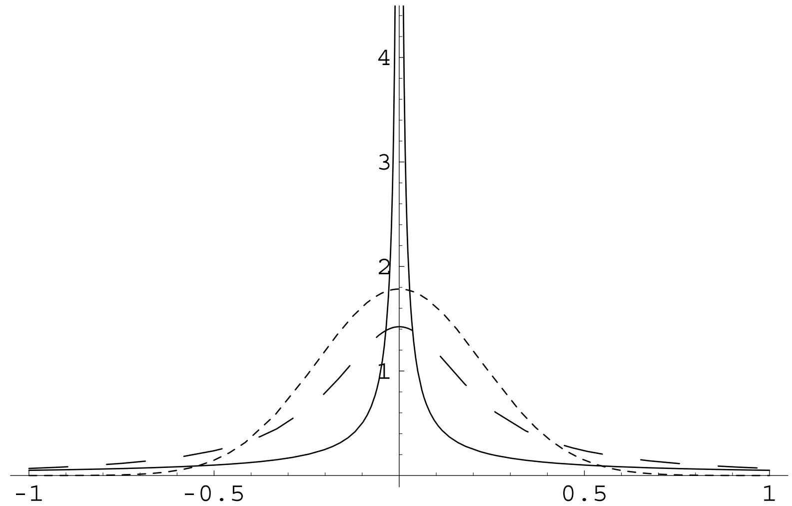

Fig. 3. The probability density function of the “Continuous-Gray” distribution for $p = 20$ (solid line). A Gaussian distribution with the same standard deviation (small dashes) and a Cauchy distribution with scale factor set to the same value (large dashes) are also shown. It can be seen that the new distribution strongly prefers small local moves, but with non-trivial probability of making larger jumps at all scales.

with the wrapping around of the search interval (inherited from the discrete Gray code) means that new points are always generated within the required bounds. Algorithms that use Gaussian or Cauchy distibutions have to be artificially adjusted when invalid parameter values are generated (either by correcting the value to be the nearest bound, or by simply resampling until a legal value is obtained).

However, it is a well-known fact that if an algorithm is good for some problems, it must be bad for others. It is therefore worth considering situations in which the new algorithm would fail to perform well. If the problem has a number of narrow, well-separated optima, then any search algorithm maintaining a single point at each iteration is likely to be trapped in one of those optima — and it will be a matter of chance whether or not the right one is chosen. It is hard to see how this could be avoided without making use of a population. A population is also helpful if one wants to introduce some crossover. This can be a good idea if there exists some correlation between the variables of the problem. We have done some preliminary investigations into using the new distribution as a mutation operator in a steady-state GA, with crossover, and we have also looked at using it in a hybrid “memetic” style algorithm, with some success. One difficult landscape feature that is harder to address is the situation where there are “ridges” running at an angle to the axes specified by the parameters of the problem (e.g. if we rotated the axes for Rastrigin’s function f_9). The most promising approach here would be to incorporate some sampling of the landscape so as to realign the search parameters with the ridges (e.g. by using a covariance mutation matrix). However, this kind of local modelling is itself quite expensive, and so we have a trade-off which may or may not be worthwhile.

References

1. Whitley, L.D.: A free lunch proof for Gray versus binary encodings. In Banzhaf, W., Daida, J., Eiben, A.E., Garzon, M.H., Honavar, V., Jakiela, M., Smith, R.E., eds.: Proceedings of the Genetic and Evolutionary Computation Conference. Volume 1., Orlando, Florida, USA, Morgan Kaufmann (1999) 726–733
2. Whitley, L.D., Barbulescu, L., Watson, J.P.: Convergence results for high precision Gray codes. In Martin, W.N., Spears, W., eds.: Foundations of Genetic Algorithms VI, Morgan Kaufmann (2001) 295–311
3. Whitley, L.D., Bush, K., Rowe, J.E.: Subtheshold-seeking behaviour and robust local search. In: Proceedings of the Genetic and Evolutionary Computation Conference. (2004) To appear.
4. Yao, X., Liu, Y., Lin, G.: Evolutionary programming made faster. IEEE Transactions on Evolutionary Computation **3** (1999) 82–102
5. Bäck, T., Schwefel, H.P.: An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation **1** (1993) 1–23
6. Hidović, D., Rowe, J.E.: Validating a model of colon colouration using an evolution strategy with adaptive approximations. In: Proceedings of the Genetic and Evolutionary Computation Conference. (2004) To appear.
7. Saucier, R.: Computer generation of statistical distributions. Technical Report ARL-TR-2168, Army Research Laboratory (2000) http://ftp.arl.mil/random/.
8. Ge, Z., Schomacker, K.T., Nishioka, N.S.: Identification of colonic dysplasia and neoplasia by diffuse reflectance spectroscopy and pattern recognition techniques. Applied Spectroscopy **52** (1998) 833–839
9. Egan, W.G., Hilgeman, T.W.: Optical Properties of Inhomogeneous Materials. Academic Press (1979)
10. Kubelka, P., Munk, F.: Ein beitrag zur optik der farbanstriche. Zeitschrift fur Technishen Physik **12** (1931) 593–601
11. Prahl, S., Keijzer, M., Jacques, S., Welch, A.: A Monte Carlo model of light propagation in tissue. In Mueller, G., Sliney, D., eds.: SPIE Proceedings of Dosimetry of Laser Radiation in Medicine and Biology. Volume IS 5. (1989) 102–111

Acknowledgements

Some of this work was done while Jon Rowe was visiting Prof. Darrell Whitley at Colorado State University, funded by National Science Foundation grant number IIS-0117209. The colonoscopy spectra were kindly given to us by Kevin Schomacker of MediSpectra, Lexington MA.