An Introduction to Evolutionary Computing for Musicians

Phil Husbands[1], Peter Copley[2], Alice Eldridge[1], James Mandelis[1]
[1] Department of Informatics, University of Sussex, UK
[2] Department of Music, The Open University, UK

philh | alicee | jamesm @ sussex.ac.uk, peter@pcopley38.fsnet.co.uk

1 Introduction

The aims of this chapter are twofold: to provide a succinct introduction to Evolutionary Computing, outlining the main technical details, and to raise issues pertinent to musical applications of the methodology. Thus this article should furnish readers with the necessary background needed to understand the remaining chapters, as well as opening up a number of important themes relevant to this collection.

The field of evolutionary computing encompasses a variety of techniques and methods inspired by natural evolution. At its heart are Darwinian search algorithms based on highly abstract biological models. Such algorithms hunt through vast spaces of data structures that represent solutions to the problem at hand (which might be the design of an efficient aero engine, the production of a beautiful image, timetabling a set of exams or composing a piece of music). The search is powered by processes analogous to natural selection, mutation and reproduction. The basic idea is to maintain a population of candidate solutions which evolve under a selective pressure that favours the better solutions. Parent solutions are combined in various ways to produce offspring solutions which then enter the population, are evaluated and may themselves produce offspring. As the cycle continues better and better solutions are found. This class of techniques has attracted a great deal of attention because of its success in a wide range of applications.

Once the Neo-Darwinian framework, which unified Darwin's theory of natural selection with genetics, had been established in the 1930s and 40s and emerged as a powerful theoretical underpinning for biology (Fisher 1930, Haldane 1932, Huxley 1942), it is perhaps not surprising that computer pioneers wondered if it was possible to abstract general problem solving methods from the logic of natural evolution. During the 1950s a number of prominent thinkers, such as Turing, suggested the use of artificial evolution as a possible methodology for developing adaptive machines. He envisioned its use in developing learning machines. Such machines would have hereditary material (artificial genes) encoding their structure, mutated copies of which would form offspring machines. A selection mechanism would be used to favour better adapted machines -- in this case those that were best at learning (Turing 1950). Such ideas were relatively common at that time, the golden age of mid-century Cybernetics when biological inspiration was rife and adventurous researchers were mapping out a visionary landscape, but it was not until the 1960s, when computer hardware became more powerful and easily available, that concrete instantiations began to appear. Three different variants independently emerged. Rechenberg and Schwefel developed Evolutionstrategies (Rechenberg 1965) to tackle engineering design optimisation problems. Fogle, Owens and Walsh (1966) describes the

technique of Evolutionary Programming, primarily concerned with evolving finite state automata for machine learning tasks. John Holland and his group at the University of Michigan developed the more general Genetic Algorithm which would become the best known of the methods (Holland 1975). Holland's early work in this area was concerned with building a powerful general formalism for adaptive systems (Holland 1962), this lead to his notion of a 'general reproductive plan' (Holland 1966) which, slightly modified, was christened a Genetic Algorithm by Bagley (Bagley 1967). It was during the 1980s that the field really took off, when it was at first dominated by work in Genetic Algorithms. In the 1990s general frameworks were developed which unified the various strands under the now widely used term of Evolutionary Computing (Back and Schwefel 1993). It is beyond the scope of this paper to look in detail at all the historical flavours of evolutionary algorithm that emerged during the development of the field (see Eiben and Smith 2003, Mitchell 1996 for good introductions), rather the main properties of such methods will be presented by appealing to the idea of a general class of evolutionary search algorithms which encompasses the major sub-dialects.

Early applications of Evolutionary Algorithms (EAs) were mainly in engineering optimization of one sort or another (see e.g. Davis 1990, Goldberg 1989, Grefenstette 1987), but as the method became better known and sparked the imagination of many researchers, the range of applications became increasingly wide and soon encompassed creative and artistic domains, including music.

The deceptively simple biological analogy at the heart of EAs is highly attractive and provides a rich seam for further developments that many researchers have mined and are still busy mining today. As we shall see, this has resulted in a highly flexible framework, with far fewer restrictions on its application than for other comparable methods, allowing plenty of scope for creative work. This is one of the great strengths of the area and one of the reasons why it is attractive to musicians and artists.

The next section gives a succinct introduction to the technical details of EAs. This is followed by sections on two particularly popular areas for musical applications of evolutionary computing: composition and sound design. Important issues arising from the use of EAs in these areas are aired. The paper then continues with a wider discussion of the place of adaptive systems in music.

2 Evolutionary Search Algorithms

2.1 Some biology

Most EAs are based squarely on the Neo-Darwinian framework from biology and borrow certain key nomenclature from it. Hence it will be helpful to outline that framework before launching into the details of EAs.

According to Darwin’s theory of natural selection (Darwin 1859), evolutionary change comes about because of the existence of variations in inheritable traits in every generation. Those individuals who survive, owing to a particularly well-adapted combination of inheritable characteristics, give rise to the next generation. The fittest survive to pass on those traits that helped to make them fit.

Darwin knew very little about how these variations arose or what the mechanisms underlying inheritable traits were. It was modern genetics that provided the key to answering these problems. Hence Neo-Darwinism postulates that natural selection acts on the genetic variations within populations – genes being the units underlying inheritable characteristics. These variations are caused by genetic processes such as mutations (sometimes caused by mistakes in DNA replication) and recombination of genetic material from different sources (e.g. the two parents in sexual reproduction).

Natural selection is usually thought of as acting on the *phenotype*, the outward expression of the genes (the *genotype*) – such as physical characteristics or behaviour, the environment it inhabits and the interactions between them. This process, together with others such as genetic drift and speciation, is a key element of modern evolutionary theory.

2.2 The basics

Figure 1 outlines the general scheme of an EA. An initial population of structures representing solutions to the problem is first created. They might be completely random individuals or based on a prior solution or generated using heuristics that ensure that they have certain desirable characteristics. Each member of the population is evaluated so that it can be assigned a fitness (usually a numerical score). This will involve decoding the genetic representation (genotype) into a problem solution (phenotype) and testing its fitness using some method for determining how well it solves the problem. Parents are selected, with a bias towards fitter members of the population, for the creation of offspring using artificial genetic operators such as mutation and recombination. The offspring are evaluated and certain of them are selected to take the place of existing members of the population chosen according to a replacement scheme (usually biased towards the least fit individuals). The cycle continues until a sufficiently fit individual emerges or some stopping criteria, such as number of cycles run, is met. Most parts of the cycle involve random, or stochastic, processes which are crucial to the success of the method.

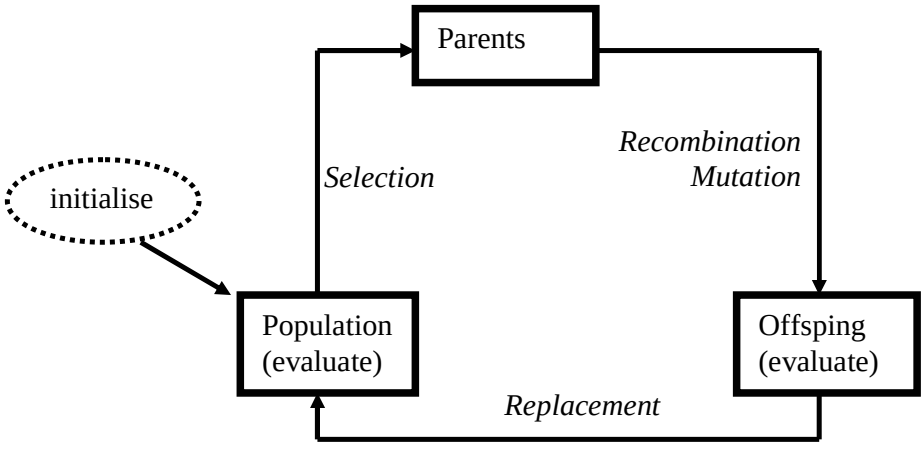


Figure 1: General scheme of an Evolutionary Algorithm

We will now look at the operation of this scheme in more detail by examining the constituent parts of the EA. The main components of an evolutionary search algorithm are:

- The genetic representation
- The evaluation function
- The population structure
- The selection method
- The genetic operators
- The replacement scheme

The genetic representation

The structures making up the population, the artificial genotypes, are usually strings of numbers or symbols that represent solutions to the problem at hand. They might be a string of real numbers that are the parameters controlling a sound synthesis method -- and hence represent a sound, or they might be groups of numbers representing information such as musical note values and duration – and hence represent a piece of music. Complex encodings involving mixtures of numbers, symbols, rules and other data structures have also been successfully used (e.g. the sub field of genetic programming is concerned with the evolution of a particular form of LISP computer programme – Koza 1992). It is also possible to use a fairly simple genotype in combination with a complex decoding scheme to translate it into the phenotype. Rather indirect routes to the end goal can be taken. For instance, the genotype may specify the design of a process, or abstract machine, which is then run to generate the end product of interest (e.g. a musical phrase). The genotypes can be of a fixed length or, where appropriate, they can be allowed to grow and shrink. The great flexibility available in designing a suitable representation is one of the major advantages over more traditional methods afforded by the EA framework. However, not all representations for a given problem will be equally good. In some cases the representation to use is fairly obvious and straightforward (e.g. a string of numbers acting as the parameters of a well defined process or design), in others it may not be so clear. The representation defines the *genotype space* through which the EA searches looking for a combination of genes that defines a sufficiently fit phenotype. If the representation is badly designed the space may become impossibly convoluted and too difficult to search with any efficiency, rendering the EA useless. Throughout the remainder of this book concrete examples of representations suitable for musical systems will be found.

The evaluation (fitness) function

EAs are a form of ‘generate and test’ algorithm (generate a new candidate solution, test it to see if it is any good), the evaluation function – which operates on the phenotype - providing the necessary means to measure fitness. As such it defines the solution requirements and implicitly encapsulates the meaning of adaptation and improvement for the particular evolutionary system. The selection method relies on the evaluation function assigning relative fitness values to members of the population in order to preferentially choose the fitter individuals to produce the next generation.

Fitness is often measured on some numerical scale, but as a minimum the evaluation function must be able to distinguish between relatively fit and unfit individuals.

The simplest form of evaluation method is a well defined mathematical function or procedure whose variable are directly encoded on the genotype; these are fed into the function and a fitness value is thrown back (Eiben and Smith 2003). For more complex phenotypes, for instance when the genotype encodes the design of a robot, evaluation often involves generating a computer model of the phenotype (e.g. the robot) and then testing its behaviour in a complex simulation (Jakobi 1998). In other cases an automated analysis of some characteristics of the phenotype is conducted in order to derive a fitness measure. For instance, an evolved musical composition might be analyzed in terms of its closeness to some target piece, or by using some musicological theory or technique (Wiggins et al. 1998). If the phenotype is the design of a physical artifact, evaluation might entail analyzing various functional and aesthetic properties of the design (Bentley 1999). In examples such as these, defining a satisfactory automated fitness measure is often highly problematic – how do we codify aesthetics, how do we formalize crucial parts of the creative process of an artist or composer? This important issue will be revisited in Section 3 and in later chapters of this book. One partial solution that is commonly used, having been pioneered in the application of EAs in visual art (Todd and Latham 1992, Sims 1991), is to employ a human's judgment as the fitness measure. The main problem with this method, sometimes referred to as *aesthetic selection* or *interactive evolution*, is the amount of time required to perform the fitness judgments. This can preclude running the evolutionary method for more than a relatively small number of cycles.

The population structure

In the simplest cases the population is just a data structure containing the genotypes and their associated information, such as fitness. The population size is often fixed, but it can be variable. In some EAs the entire population is replaced on each cycle, which is then referred to as a generation. A more sophisticated variety of EA uses a *spatially distributed* population, alluding to the underlying conceptual model of the population spread out over a 2D grid with each individual occupying its own cell. Members of the population only interact with those individuals sufficiently close to be in their *neighbourhood*. Hence selection and reproduction act asynchronously and locally allowing for highly parallel implementation of an EA (for instance using a network of processors, one for each cell on the grid). This form of EA has been shown to be highly efficient (Collins and Jefferson 1991, Hillis 1990, Husbands 1993).

The selection method

Selection, whereby more credence is given to fitter population members, provides the dynamo that powers the algorithm. The fittest are *more likely* to pass on some of their genes to later generations. This probabilistic element - which is found in other parts of the method, for example the genetic operators - helps to account for the technique's power and robustness.

A simple and reasonably effective selection method is **roulette selection.** In this scheme each member of the population is assigned a probability of selection based on its relative fitness (its fitness value divided by the total population fitness). Parents are

then selected according to this probability. This is analogous to dividing up a roulette wheel into N sectors, one for each member of the population - sizing them according to the relative fitness of the individual represented, and then spinning it to select parents. The bigger the relative fitness the more likely the individual is to be selected for breeding. Note that with this scheme no member of the population is excluded from breeding, they all have some chance of contributing to the next generation. However, this method can result in too strong a selective pressure in favour of individuals that are relatively good at the early stage but may actually be far from optimal; the population *prematurely converges* to be dominated by copies of such individuals.

Rank-based selection is a particularly straightforward alternative scheme that provides more control over the selective pressure and allows strong differentiation of the population, even at later stages when their fitness values are very close. Using this strategy the population is ranked, or ordered, according to the fitness values of its members. Selection is then performed by following a pre-determined probability distribution function, such as the ones shown in Figure 2. This may be a simple linear function that constrains the first ranked (fittest) individual to be twice as likely to be selected as the median ranked individual, or something more complex.

An alternative form of selection, that makes most sense in the context of parallel EAs, was alluded to earlier - the use of **local selection rules**. Briefly, the idea is that a population is somehow split up into many subpopulations, either explicitly or implicitly (as in the case of the spatial distribution mentioned above), and selection occurs *locally*. That is, with reference only to the subpopulation, not to the global population. Local schemes may be based on the methods described earlier or may be simpler. For a more detailed discussion of possible selection schemes see (Eiben and Smith 2003, Mitchell 1996).

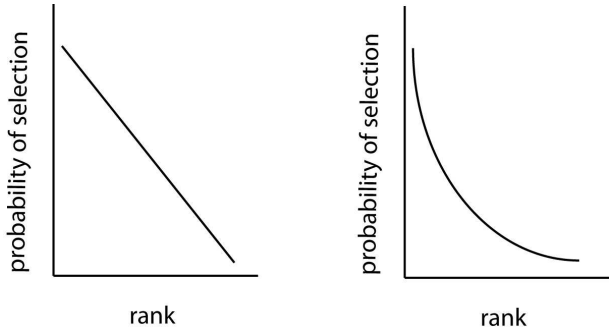


Figure 2: typical rank-based selection probability distributions

Interactive evolutionary algorithms, employing human-based aesthetic selection, effectively dispense with a separate selection method – individuals are picked out by the user to act as parents for the next generation.

The genetic operators

The genetic operators maintain variation in the population and create new individuals from old ones. Myriad specialised operators have been developed over the years and

there are numerous variation on the standard ones. Hence only a few of the most common generic operators will be outlined here. The two most common are **cross-over** and **mutation**, illustrated in Figure 3. Like most widely used operators, they have strong stochastic elements to their operation. Simple cross-over involves choosing at random a cross-over point (some position along the string) for two mating chromosomes -- two new strings are created by swapping over the sections lying after the cross-over point. Variations include two-point cross-over where randomly selected sections of the strings are swapped over and special operators that rearrange genes during the crossing-over, either in order to keep the new solutions legal or to make them better (Michalewicz et al. 2004). Mutation changes the value of a gene to some other possible value. Depending on the encoding, this might entail assigning a new value at random from the entire range of possible values for the gene, or randomly resetting to a value 'close' to the current one (*creep mutation*). Mutation operators can be heuristically guided, rather than completely blind (e.g. if a gene represents a note value in a piece of music, mutation operators might be designed to respect certain harmonic or melodic constraints – or perhaps more interestingly to *nearly always* respect them). For complex encodings, it often makes sense to have several different mutation operators acting in parallel. Other operators sometimes used include: **inversion,** which is simply a matter of reversing a randomly chosen section of a single genotype, **translocation** which involves moving a randomly selected section to another place on the genotype, and **duplication** which entails adding extra copies of genes or groups of genes. The latter operator only makes sense in circumstances where a variable length encoding is being used, it often functions in tandem with a **deletion** operator. Specially designed cross-over operators can also be used to allow genotypes to grow and shrink (Harvey 1992). Special domain specific operators are regularly employed to good effect. For instance, in the application of EAs to musical composition operators based on musical transformations such as inversion and transposition can be very useful (Biles 1994).

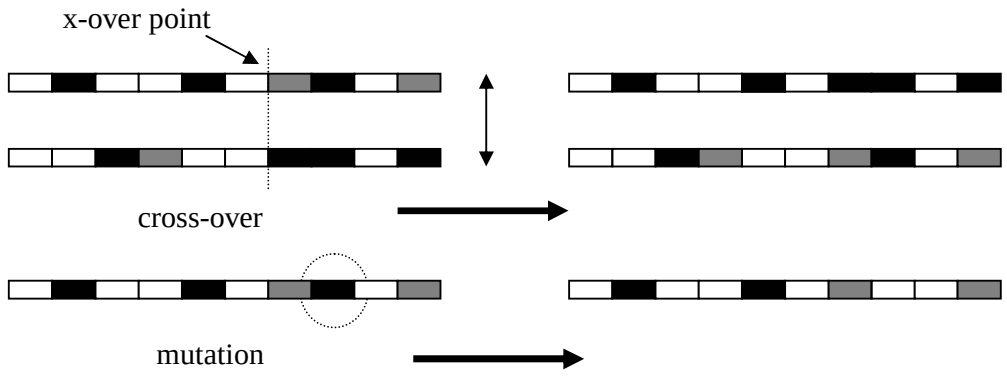


Figure 3: Schematic of popular genetic operators

The operators have assigned *rates* which determine how likely they are to be used. They are applied at the offspring creation stage according to a routine like the following. When two genotypes are selected for breeding, first apply crossover (with

some high probability) to create two new genotypes. Next apply inversion (with a medium probability) to these. Finally each gene on the resulting genotypes undergoes mutation (with a low probability). According to the encoding scheme and problem area, different combinations of operators with different rates are used. In some circumstances it makes sense to dispense with cross-over, for instance if it is difficult to devise an encoding that works with this operation, and just use one or more mutation operator. It is common to have to experiment with operator rates to find good settings, which can usefully be made to vary during the search – in some cases the rates themselves are put under genetic control (Back et al. 1991).

The replacement scheme

In some EAs enough offspring are produced on each cycle to replace the entire population in one go. In others, sometimes called *steady state* algorithms, new individuals are introduced one at a time, as long as they are fitter than at least the worst member of the population which is then replaced. This allows a more gradual search. Other schemes use an inverse selection method to choose members of the current population to be replaced with a bias towards the least fit.

It should be clear from this brief outline of EAs that there are many choices to make in deciding how to apply them to any given domain and many parameters to tweak once the basic algorithm has been designed. The various elements of the EA must all work well together in order to achieve good results. The best choice of operators, genetic representation, evaluation function and so on, can be guided by what has been shown to work in the past, or by experimentation with different settings and options. However, some appreciation of the growing theoretical understanding of how EAs work can be very helpful and save time spent down the blind allies of poor representations or inadequate fitness functions. EAs are complex non-linear stochastic systems, which makes them extremely difficult to analyze. Hence the theoretical literature tends to be rather inconsistent and is often contradicted by empirical results. However, there is useful information to be gleaned and good sources include: (Schmitt 2001, Vose 1999, Wright et al. 2005).

2.3 Related developments

Evolutionary algorithms have played an important part in the development of the related fields of Artificial Life, which is concerned with the synthesis and analysis of life-like processes in artificial media (Langton 1995, Pollack et al. 2004), and Adaptive Behaviour, which studies the mechanisms underlying the generation of adaptive behaviour in real and artificial autonomous agents (Beer 1990, Schaal et al. 2004).

These areas have seen interesting explorations of phenomena and techniques that have found applications in artistic endeavours. For instance, coevolutionary systems, in which two or more ‘species’ compete (or possibly cooperate) over finite resources, have been exploited in Karl Sims’ entertaining animations, in which primitive creatures wrestle with each other (Sims 1994), as well as in various engineering applications (Husbands 1993, Juille and Pollack 1996). This is a direction which

might hold some promise as far as computer music is concerned – multiple species could represent different voices in a composition or, following the suggestion of Werner and Todd (1997,1998) to mimick work such as (Hillis 1990), and not withstanding the issues raised later in Section 3, it might be possible in some situations to have coevolving species of compositions and 'critics' (who evaluate the compositions) developing towards some interesting end.

Jon McCormack's Eden is an interesting example of an installation using ideas from these areas (McCormack 2003). In this system, simulated beings populate an artificial world in which they can move around and make and hear sounds. These sonic agents must compete for limited resources in their environment. The agents generate sounds to attract mates and also to capture the imagination of the audience - since its response has a direct affect on the virtual environment, particularly the growth rate of food. In this work McCormack has demonstrated the successful use of an open-ended automatic evolutionary process to generate a highly engaging interactive artwork. This system illustrates a more implicit approach to fitness evaluation, with a fairly oblique interaction element. Such pieces suggest a wealth of opportunities for musical developments.

2.4 Applications of evolutionary computing in music

There is a growing body of work involving the use of EAs in musical applications (see Horner and Goldberg 1991, Burton and Vladimirova 1999, Bilotta et al. 2001, Miranda 2003 for representative examples), just as there is in the visual arts and in design (Bentley 1999). In music the two areas that have attracted the most attention are composition and sound design. In the former, there have been a number of attempts to evolve musical pieces in the style of a particular composer, or within a specific idiom, which have met with some success (Biles 1994, Hodgson 1999, 2002) – although extending such work to more creative and original compositions is challenging, for reasons including those discussed in Section 3. In the area of sound design, researchers have demonstrated the efficacy of the technique in controlling sound synthesis methods, both to explore new sounds and to develop synthesis algorithms for existing target sounds (Johnson 1999, Dahlstedt 2001, Garcia 2001, Mandelis 2001).

Various aspects of these topics, in relation to specific systems, will be dealt with in detail in later chapters. The remainder of this chapter is intended to raise a number of important issues in these areas as background and context to the rest of the book. Fitness evaluation turns out to be a particularly thorny issue in relation to compositional systems and it is not a trivial matter in sound design.

3 Evolutionary Computing in Musical Composition

3.1 Introduction

The main purpose of this section is not to attempt a comprehensive survey of evolutionary computational approaches to musical composition (see Burton and Vladimirova 1999 for a good overview, as well as later chapters in this book) but

rather to highlight some of the potential problems, apparent in the literature, of too close a marriage between the development of compositional computer programmes and an approach to musical form derived primarily from academic theory, rather than what many composers demonstrably do. This is a very real problem as textbook musical form is by its nature algorithmic and has often been seen as the ideal starting point for the development of composition programmes, particularly those based on pre-existent models of compositional practice (see Wiggins *et al*. 1998). The main 'test case' for discussion in this section will be sonata form, in theory and practice as this, in particular, is a type of composition that could well prove problematic if the creative process to be modelled is not based on a traditional textbook definition but rather, something paralleling an end product that significant composers actually produced. This is not to suggest that the production of an interesting sonata structure per se is a primary goal of more than a minority of practitioners in this field. Rather, that sonata form itself was a significant tool (whether algorithmic in nature or not) in the evolution of complex musical structure for more than one hundred and fifty years in the history of Western Art Music. Its potential to encompass so many elements that inform the creative process – exploration, contrast, development, transformation, motivic mutation etc. – make it an ideal context within which to examine the potential limitations of EA composition programmes.

The musical forms generated by an EA-based system will be implicitly restricted and shaped by the design of the various components of the system – most importantly the genotype, the genetic operators and the fitness function. If an automatic fitness evaluation method is used, the desired musical outcome must be somehow formally codified. Deriving sets of rules to describe particular forms or styles is fraught with difficulties, as discussed below. If the automatic fitness function problem is sidestepped by using human evaluation, the search space defined by the genetic representation and operators must be sufficiently constrained to avoid impossible bottle-necks in the time needed to perform the evaluations (Biles 1994, Gartland-Jones and Copley 2003, 2005). As this will entail encoding musical knowledge into the representation and operators, the difficulties do not disappear.

3.2 Algorithmic composition

In 'The Algorithmic Composer', David Cope stated that throughout the history of Western Art Music, composers have used algorithms as part of the creative process. His premise was that an algorithm could be defined as nothing more than 'a set of rules for solving a problem in a finite number of steps' (Webster 1991, p.35; cited in Cope 2000, p.1). Clearly this is of crucial importance to anyone engaged in building AI models of musical creativity and assuming Cope's premise is valid, algorithms of musical composition and form building would be central to the construction of such models. However, while it is perfectly possible to define some compositional processes as algorithmic, not all fall so neatly into this category. A necessary preliminary step would be to attempt a delineation of boundaries, as to what extent, which compositional processes can and which cannot be so defined.

Cope stated that 'Most composers apply rules, steps, or sets of instructions when composing music, especially when composing music in a particular style' (Cope 2000, p.2). Part of his support for this proposition is a series of examples of

compositional processes defined as algorithmic. These include the tenor part of an isorhythmic motet (significantly, Cope omits discussion and illustration of the other voices, which are freely composed), Bontempi's *rota*, musical dice games and Fux's *Gradus ad Parnassum* (Cope 2000, pp.3-11). This is a wide-ranging set of examples although, with the exception of the motet, all bear only a peripheral relation to musical composition as actually practised by fully-fledged composers. Bontempi (1660) states that his *rota* is a guide 'by means of which one thoroughly ignorant of the art of music can begin to compose'; a sort of musical equivalent of 'painting by numbers', in point of fact! Musical dice games were similarly 'do it yourself kits' for beginners, while Fux's *Gradus* (1725) was the standard instruction book for learning strict counterpoint for much of the eighteenth century – a useful tool for the elementary technical training of aspiring composers but bearing about the same relation to real music as a book of finger exercises, however advanced, would have to the performing repertoire of a professional concert pianist.

On the subject of form, Cope writes that:

> "Strict adherence to an established musical form constitutes yet another compositional use of musical algorithms. For example, imagining a song form of the medieval period, a dance form of the baroque, or a sonata allegro form of the classical period of Western music history as symbols in a flowchart – one way to describe an algorithm – does not seem unreasonable."(Cope 2000, pp.3-4)

The problem arises with Cope's un-stated but implied assumption that significant composers at all periods in the development of Western art music did indeed adhere strictly to *established* musical forms in their most original work, even granting that these forms were already in acknowledged existence at the time of writing, rather than being deduced after the event by historians or writers of textbooks on musical composition!

To return briefly to Fux, it is of course documented that Haydn, Mozart and Beethoven, to name but three, worked assiduously through the exercises in the *Gradus* or from textbooks of a similar nature; but it is equally demonstrable that they paid scant attention to the *letter* of the majority of Fux's rules in their compositional maturity. This is not to suggest that Fux is valueless as an example of a producer of musical algorithms, simply that the process of modelling anything more than the most elementary compositional process is rather more complicated than his citation by Cope might suggest. Historically, a large claim made for the benefit of strict counterpoint study of the Fuxian variety was that it provided what amounted to an algorithm for composing in the style of Palestrina, who had been regarded for centuries after his death as a byword for purity of contrapuntal style. Unfortunately, this claim was largely unfounded and was completely exploded by R.O.Morris as far back as the nineteen twenties:

> "Yet the rules of Mr Rockstro [*another author of a book on strict counterpoint*] are not peculiar. They are, more or less, the same as those found in almost every textbook of counterpoint. Who invented them, goodness only knows: why they have been perpetuated, it passes the wit of man to explain. Music written to meet their requirements is something altogether *sui generis*, a purely academic by-product…The rules of counterpoint are found to have no connexion with

> musical composition as practised in the sixteenth century: are we to abandon the rules or to abandon the sixteenth century? Follow Byrd and Palestrina, or follow Mr.Rockstro and Professor Prout?"(Morris 1922, p.2)

3.3 Is sonata form an algorithm?

Sonata form expressed as an algorithm brings similar problems in its wake. Is the algorithm to be based on textbook definitions or on what significant composers actually produced? Furthermore, there remains the question of which variety of sonata form as practised is to be taken as the starting point. Many commentators (see, for example, Rosen 1980, pp. 365-402 and Straus 1990, pp.96-7) are now in agreement that there exists a fundamental distinction between what could broadly be described as eighteenth and nineteenth century approaches. For a composer in the second half of the eighteenth century, sonata form (not termed as such) was an elaborated binary structure characterised by differentiated key areas. The first part contained a tonic area and a dominant (or related key) area, although the first area could be characterised by a modulation to the tonality of the second area. The second part consisted of an area of rapid modulation or episode followed by a return to the home key in which tonality the movement remained until its end. The two-part view of sonata structure is confirmed by the prevailing eighteenth century practice of repeating both sections, rather than just the first part, as is usually the case in contemporary performance.

What is set out above is just about the fullest extent of universal common ground in composing practice that can be extrapolated from the majority of later eighteenth century sonata structures and a composing algorithm extracted from this would be little different from one derived from baroque binary dance patterns, despite the two forms being in reality quite distinct from each other. The distinction between the two is the far greater proliferation and elaboration of material that the sonata framework came to accommodate – what could, in fact be termed 'free composition'. The beauty of the form lay in its flexibility. This minimum common ground, never at this stage delineated in any contemporaneous textbook on composition, could accommodate not only Haydn's largely monothematic and developmental approach, but also Mozart's, which tended to explore the underlying unity of two or more distinct but nonetheless contrasting themes.

All this came to change in the nineteenth century, thanks initially to the theorising of A.B.Marx (1795-1866) and Karl Czerny (1791-1857), which was largely based on the sonata practice of 'middle period' Beethoven, who had provided yet another distinct approach to the original but still evolving model. I give here Schoenberg's description of the form, which corresponds to the nineteenth century theorists' view, which was concerned less with the delineation of key areas and more with thematic contrast, expressed in a ternary rather than a binary context:

> "This form…is essentially a ternary structure. Its main divisions are the EXPOSITION, ELABORATION and RECAPITULATION. It differs from other complex ternary forms in that the contrasting middle section (ELABORATION) is devoted almost exclusively to the working out of the rich variety of thematic material ''exposed'' in the first division. Its greatest merit, which enabled it to hold a commanding position over a period of 150 years, is

> its extraordinary flexibility in accommodating the widest variety of musical ideas, long or short, many or few, active or passive, in almost any combination. The internal details may be subjected to almost any mutation without disturbing the aesthetic validity of the structure as a whole."(Schoenberg 1967, p.200)

Although Schoenberg proves himself a child of the nineteenth century in his thematic and ternary, rather than tonal and binary, view of sonata structure, perhaps in order to allow for his continuing to explore the form in non-tonal contexts, his description is still loose enough to accommodate a wide variety of approaches, including those of the later eighteenth century. The compositional algorithm that could be extrapolated from this description would, however, differ little from one derived from a simple ternary form.

For a truly distinctive sonata algorithm, resembling neither the simple binary nor the ternary model, we would need to turn instead to the traditional theorists, who would state that Sonata Form consists of firstly, an *exposition*, comprising first and second subject groups, respectively in the tonic and dominant (or related) keys and linked by a *transition* or *bridge passage*; secondly, a *development section*, in which the original thematic material will pass through a variety of related keys and may be extended by *episodes*; this will be followed (thirdly) by a *recapitulation*, in which the material from the exposition returns but is mostly confined to the original key. Various optional extras, such as *introductions*, *codettas* and *codas* can fill out the scheme and may be represented as byways on a flowchart, which is Cope's preferred method for setting out compositional algorithms in a non-computerised context.

Actually, this theoretical description does indeed correspond to more conservative later nineteenth century practice and this lends a depressing sameness, from a purely formal point of view to the majority of sonata-type structures from this period. The extraordinary paradox is that the 'romantic' nineteenth century was far less free than the 'classical' late eighteenth century in its interpretation of what might be termed 'the sonata principle', except in the case of more progressively minded composers, such as Liszt, Berlioz and Wagner, who tended to abandon the form completely. It is difficult to avoid the conclusion that once 'the rules' had been encapsulated in a detailed formal scheme or algorithm, the sonata began to lose its dynamic and developmental possibilities and its various sections took on the character of moulds into which appropriate music could be poured. Such an approach to potential sonata material would have been psychologically impossible for any major eighteenth or early nineteenth century composer of whose structure-building creativity generally went beyond simply following formulae devised by others.

3.4 The dangers of too many and too few rules

It may seem that several of the preceding paragraphs address issues more central to the concerns of musical historians, analysts and aestheticians than those of designers of computer programmes for musical composition. However, if we are modelling musical creative processes to any degree of sophistication, it is crucial that we base our model on something close to what composers actually did, rather than on theoretical constructs, often established long after the creative event, that oversimplify or distort complex thought processes in the interests of pedagogical expediency. An

excessively rule-based system stands in grave danger of producing little more than schoolroom exercises or, at best, stolid replications of 'good craftsmanship' because no facility has been provided for expanding a given search space to accommodate the possibility, indeed the desirability of the unexpected, or even iconoclastic but still meaningful musical idea or development.

Although the explorative and stochastic nature of evolutionary search are helpful, this is perhaps the most challenging problem facing the designer of an EA-based composition program, whether for general use or tailored to one particular set of preferences. The past decade and more has shown that an EA has no difficulty in replicating a composer in 'hard-work' (as opposed to 'inspired') mode (See Jacob 1996: 158). But without the most stringently defined search space an unmanageably large amount of potential material, mostly unusable, is apt to be produced. Biles (1994) has described this situation as the fitness bottleneck. However, if the search space is too strictly defined – 'Strict adherence to an established musical form' (Cope: 2000: pp3-4))- the unexpected and interesting permutation, which is what all the hard work is supposed to uncover may not emerge at all. As Werner and Todd pointed out:

> "More structure and knowledge built into the system means more reasonably structured musical output; less structure and knowledge in the system means more novel, unexpected output, but also more unstructured musical chaff." (Werner and Todd 1998:315)

What algorithm from textbook musical forms could have allowed for Haydn's unprecedented departure from the expected course of musical events in the developmental extended coda that erupts into the final variation on a theme in the slow movement of his String Quartet, Op.20 no.4; Mozart's introduction of a modulatory and developmental theme that is *not*, contradicting all expectation, the second subject of the first movement of his *Haffner* Symphony; Beethoven's 'sonata structure, accommodating variation' (Keller 1987, p.136) that forms the choral finale of his 9th Symphony; or Schubert's fusion, by thematic integration of the four movement sonata scheme into a single continuous movement in his 'Wanderer Fantasy'?

These are not isolated, eccentric examples but the essence of a truly creative use of form, wholly characteristic of their respective composers, which can lend musical compositions their enduring power to fascinate and hold the attention. It is this capacity to reinvent (or, particularly in the case of Haydn, to invent) form that is a fundamental difference between a Haydn and a Vanhal; a Mozart and a Dittersdorf; a Beethoven and a Diabelli; or a Schubert and a Hummel. Meaningful contradiction of expectation is one expression of individuality that distinguishes specific pieces and composers from the more typical cultural products of whatever age in which they lived, giving the music an intrinsic value that can transcend time and place.

To attempt to model this level of creativity is asking much of a process still in a comparatively early stage in its development but it seems vital that the possibility of overriding 'rules' must be provided for in composition programmes with any pretensions to model creative, rather than reproductive musical thought. The historical fact that theory so often followed, and in the process distorted, practice should in itself be warning enough of the pitfalls of regarding compositional processes purely as

algorithms. It is natural to have recourse to algorithms when modelling creative processes, as every computer programme ever devised is in essence algorithmic. However, it must also be recognised that if the algorithm employed is reductive and constricting in relation to the process it is modelling, the musical interest of what emerges will be at best limited, if not utterly predictable.

3.5 IndagoSonus

Gartland-Jones' *IndagoSonus* System is a very interesting approach to partially addressing some of these issues (Gartland-Jones 2003). The system uses virtual blocks which each have the ability to both play and compose music. As the blocks are arranged in various structures they interact with each other in ways that influence the emerging music. Each block has a pre-composed 'home' musical phrase and the ability to compose new phrases based on its home phrase and a phrase that is passed to it from another block. A block's compositional activity is aimed at producing a new musical section that has a thematic relationship to both of these pieces. To do this it uses an EA which is initialised with the home music and has the incoming phrase as its compositional target, allowing the use of an automatic evaluation function that measures the closeness of fit to the target. The path taken by the EA generates intermediate material related to the home and target pieces. The user can stop the evolutionary processes at any stage and restarted it with new incoming phrases, as well as setting parameters that control how far the evolutionary process will travel between the two pieces. To quote the designer:

> "Any number of blocks may be chained or grouped in any 3D structure. If a block is passed some music from its neighbour, it first recomposes itself, and then passes its new music on to all of its neighbours, and so forth within a pre-specified range. It is important to clarify that each block holds on to its home music throughout, enabling any music composed by it to remain thematically related, despite the constant process of re-composition undertaken by each block. In this way the composer of the music for all blocks maintains a compositional thumbprint on the evolving musical structure. In effect, the listener/performer is able to shape the overall music by choosing to send musical fragment from blocks they like to influence other blocks." (Gartland-Jones and Copley, 2003, p.53)

By this subtle mixing of automatic fitness evaluation and human intervention, not to mention the use of multiple interacting EAs, the system makes some headway in addressing the fitness bottle-neck problem while avoiding over constraining the search space.

4 Evolutionary Computing in Sound Design

The use of EAs at the sound level is concerned with the manipulation of parameters that define a sound using a particular sound synthesis technique (SST), or with parameters that define a particular deformation on an input stream (sound effects). There are two broad categories of EA application in this area: as an optimisation technique for deriving the parameters of an accurate model of a particular sound (usually a sampled sound) and for exploratory search in the investigation of new

sounds. These areas are briefly introduced in this section while highlighting pertinent issues.

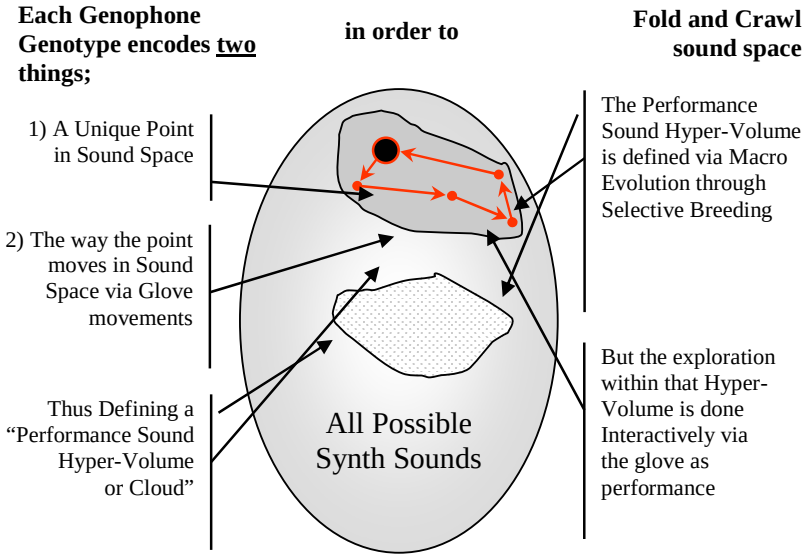
In the optimisation case a sample of sound, often from a traditional instrument, is used as a target waveform. An EA is put to work to derive the parameters of a particular SST to produce a sound as close as possible to the target. A fitness function that measures the difference between a candidate sound and the target is usually employed and there are many technical issues involved in how best to define this. There are a number of examples of successful uses of this approach (e.g. Garcia 2001) and it will be looked at in more detail later in the book. Sound definitions usually describes a singular point in the parameter space of the SST without explicitly detailing how this sound changes and deforms from that frozen point. Such deformations of sound, or movements in parameter space, are necessary for mapping the sampled instrument to a keyboard and note scale and implementing other transformations that add expressivity to the sound. In order to map those dynamics from the original source of the sampled waveform, generally a large number of waveforms is needed. As an absolute minimal requirement, at least three distinct waveforms would have to be used for each degree of freedom of the original sound source. For instance, if the source is a piano sound, the degrees of freedom of the piano would include: the key position, velocity, aftertouch and so on. In practice most instrument sounds do not vary in a linear fashion along their axes of freedom and far more than three samples would have to be used for each axis. That can very easily result in a prohibitive number of samples that places too high a computational demand on the EA. This can be a serious problem only if this technique is used to faithfully emulate an original sound source. In contrast, if such fidelity is not required, then some interesting possibilities begin to emerge. For instance, if the specific parameters are derived from a single waveform, then any deviation from these parameters will create sounds that are similar to the original but with deformation characteristics that depend on the particular SST used. For example, if a piano sound is used to derive the parameters for an FM SST and a physical modelling SST, then the deformations afforded by the former would be unique to this particular implementation of FM and for the latter unique to the particular physical modelling used. In effect there would be two instruments that would sound very similar at some performance configuration, but at the same time they would behave very differently in terms of sound deformation when the performance configuration changes.

The second category of EA-based sound creation, that of developing new sounds, requires a somewhat different approach. Whereas there is a large body of knowledge that can at least act as a starting point in attempting to formalise the evaluation process of EA-based composition systems, in the area of new sound design, where the ‘quality’ of the sound is to be assessed, there is no equivalent source. This is partly because of the complexity and lack of transparency of SSTs and partly because of the difficulty in modelling aesthetic judgements. In this domain the subjective usually rules over the objective. Hence the use of human-based interactive selection is the norm (Dhalstedt 2001, Mandelis 2001, Mandelis and Husbands 2003, Yee-King 2000, Woolf 1999), which raises the issue of the evaluation bottle-neck already discussed in relation to composition. Although there are general problems such as maintaining a consistent judgement of quality, the time taken to evaluate a sound is usually considerably less than that for a composition. This means that it is often feasible to run the algorithm for a reasonable number of cycles. The less constrained approach

necessitated by the lack of formalised knowledge allows for a powerful exploration of sound space; the user is free to navigate a world of sonic possibilities, turning up interesting and unexpected new forms that can be put to good artistic use.

Genophone (Mandelis 2001, Mandelis 2002) is one such exploratory system, designed in part to allow a flexible exploration of sound spaces without the need for detailed understandings of SSTs. Aspects of the system will now be briefly described, focusing on general issues in the way evolutionary search is used. The system makes strong use of genetic recombination which in biological systems is a creative process in itself. A biological analogy would be the breeding of animals or plants which humans have done for millennia. When pigeons are bread, for example, it is not normal (at least not yet) to employ gene level manipulations via genetic engineering. Instead, macrocosmic manipulations such as artificial insemination or pair choices are enough to manipulate the genome as a whole and consequently the resulting offspring. Genophone provides analogous macroevolutionary manipulations to those employed in organic breeding: parents can be selected by the user, particular traits can be encouraged and manipulated. In addition, via dataglove manipulations, it provides a local direct and interactive exploration that facilitates smaller changes when used as a performance tool.

The issue of an instrument's degrees of freedom and the movement in this parametric space as "performance" (Pressing 1990, Rovan et al.1997, Wessel and Wright 2000, Mulder 1994) was considered as an integral part of an instrument's (sound) definition during the design and implementation of the Genophone system. This was achieved by evolving the particular parameter values that produce a desired sound *along with* a performance mapping scheme where a subset of those parameters is mapped onto manipulation devices (dataglove and keyboard controls) for use in performance, this is illustrated in Figure 4.



Thus implementing both discontinuous (macro) and continuous (local) modes of sound search

Figure 4: exploration of sound and performance mapping spaces with Genophone (after Mulder 1994).

The option of locking individual genes, or even whole sections of the genotype, provides an added layer of control over the evolutionary process that helps bridge the gap between a totally free-form search and the tight regulation offered by a manual sound editor. The inspiration for parameter locking came from the way genes are activated and deactivated in biological genomes, producing epigenetic evolutionary effects (Singh and Krimbas 2000).

An important difference between the way EAs are generally used in constrained searches towards fixed sound targets, on the one hand, and unconstrained exploration of sound spaces, on the other, is the choice of initial population. In constrained EAs a population of random individuals is often used to jump-start the evolutionary process. This is partly to ensure no initial bias exists which may direct the search away from the global maximum -- the perfect match to the target. In the unconstrained exploratory case, this is not necessary; in fact experiments with Genophone have shown that it is not even desirable. These experiments indicated that it is preferable to seed the initial population with sounds that have been professionally hand-designed and are of some aesthetic quality. A large amount of knowledge is embedded in the parametric definitions of these sounds, information that ultimately encodes a set of aesthetic values, albeit in an implicit and not easily decipherable way. By using such hand-designed sounds as points of departure for the evolutionary search, this embedded knowledge can be exploited. Experiments with Genophone also revealed that starting from hand-designed origins does not necessarily mean that the resulting offspring would sound very much like their parents. In fact sometimes they can sound surprisingly dissimilar, yet somehow still retaining some of the original quality of the hand-designed parents. It is also very easy at each generation to ‘mate’ a preferred offspring with a newly chosen hand-designed sound, thus rapidly diverging from the original parent set.

These two distinct uses of evolutionary computing for sound design have been described as the “survival of the fittest” and as the “survival of the prettiest”, drawing an analogy with the biological processes of natural selection and sexual selection. This first is in essence a convergent process whether the second one is divergent.

5 New Musical Possibilities Through Adaptive Systems

5.1 Introduction

The themes of the previous two sections are broadened out in this section into a discussion of evolutionary and adaptive algorithms as tools for exploring new musical possibilities. In particular, it will be argued that adaptive systems can provide a rich interactive mechanism for performing as well as composing with the computer.

Musicians have always made use of, and arguably inspired, new technologies. The computer opens up an unimaginable scope for developing new sounds, new aesthetics and new composition and performance practices. Audio development programmes and languages such as MAX/Msp, PD and SuperCollider are broadening the community of computer music composers, and making the implementation of systems for exploring new musical possibilities easier and quicker. The challenge, of course, is to make something that anyone actually wants to listen to. Early computer music composers revealed formidable new worlds of acoustic textures that were impossible to achieve with acoustic instruments, but it has been suggested that the diminished audiences for 'serious' computer music may be associated with an over zealous enthusiasm for precise and elaborate formalisms (Garnett 2001). Miranda has suggested that part of the problem for listeners is that these formal systems 'lack the cultural references that we normally rely on when appreciating music' (Miranda 2003, p.1.) But although lacking the hallmarks of any particular catalogued musical tradition, the organisational structures of the dynamics of some evolutionary and adaptive systems bear strong similarities with the morphologies and structures which appear across all musical styles. The behaviours of some models have an inherent liveliness that has been shown to effectively mimic certain musical phenomenon, and exhibit complex structural dynamics that have been shown to be musically effective at all levels, from timbral morphologies to long term structure at the level of musical form. In addition, the responsive nature of some adaptive systems offers an appealing mechanism for interactive performances allowing us to integrate the aesthetically challenging possibilities of computer music within the traditions of human performance practice.

5.2 Generating Structure in Time

Superficially, an evolving population of digital genes may seem to have little in common with our concept of musical form. But this model of artificial evolution shares with music a very fundamental characteristic: it is a temporal process. That it exists in time is one of the few uncontroversially universal features of music, yet consideration of dynamic form is rarely a primary consideration in computer-assisted composition. A common problem reported by practitioners of computer-based approaches (such as rule-based systems and neural networks as well as some evolutionary systems) is that despite successfully creating specific elements, there is a lack of overall musical energy or flow. For example, while discussing constraint-based system for harmonisation Lischka notes: "The harmonisations are (in a sense) correct. But they are not exciting. What is lacking is some kind of global coherency." (Lischka 1991, p.237). This makes the creation of long term or hierarchical structure a real difficulty. It seems likely that these problems are associated with the fact that time-based structures are rarely a primary focus, a tendency which perhaps has deeper roots in the music theoretic principles from which many models are derived (for a discussion of the temporal paradox in musicology see Cook 1990).

There are myriad time-based models that could be used for generating music, and many composers have explored their possibilities. The fact that a process is formally defined as a function of time does not in any way ensure that the musical outcome will be engaging, nor even that the temporal dynamics can be appreciated by the listener. Just as the application of EAs demands careful formulation of representation

schemes, fitness functions and operators, this approach relies on the inspired selection and implementation of a suitable model and the definition of a meaningful mapping from numerical output to musical space.

The implementation of a model is often motivated by an intuition that it shares an organisational structure with a particular musical phenomenon or effect. The musical success of the approach is then dependent upon mapping the numerical output into a suitable musical domain in a way which preserves the desired structure. In a later chapter in this book, Miranda describes various implementations of Cellular Automata (CA) models for musical applications. In one of these, *Chaosynth*, a chemical oscillator CA is used to parameterize a granular synthesis engine (Miranda 2000). The dynamics of the chemical oscillator CA rule, as it evolves from a random state to sustained oscillation, bear strong resemblance to the morphological evolution of sound in the voice and many acoustic instruments: their partials converge from a random distribution to a stable pattern of oscillation. The mappings used to parameterize the granular synthesis engine preserve these characteristics, so the sounds produced similarly bear these morphological features, capturing the *global* spectral evolution of an acoustic note onset. Using a complex dynamic model allows the description of the changes in amplitude of multiple frequencies over time, and also the relations between them. These multiple levels of related dynamic structures are not peculiar to the timbral level, indeed almost all polyphonic music can be conceived as a complex of distinct, but interdependent voices weaving spatio-temporal forms at many levels. The use of complex dynamic systems enables the generation of these sorts of rich spatio-temporal structures which are seen at all levels of musical organisation.

As well as modeling musical form, specific musical phenomenon can be modeled using time-based systems which would be difficult or impossible to capture using other approaches. Tim Blackwell's work on Swarm music (also presented in a later chapter in this book) is motivated by the similarity between the self-organisation exhibited by the swarm algorithm and the self-organisation, or structure that emerges in improvised ensembles of live musicians:

> "The development of higher level musical structure arises from interactions at lower levels, and we propose here that the self-organisation of social animals provides a very suggestive analogy." Blackwell and Young 2004, p.137.

The swarm system used by Blackwell is an extension of Craig Reynolds 'Boids' algorithm (Reynolds 1987) which mimics the behaviour of a flocking birds. In this simple model, Reynolds shows that the global organisation of the flock can arise from simple rules which determine the movements of each bird relative to each other, without the need of any 'leader' or pre-devised plan. The model is based on three simple principles: separation, alignment and cohesion. Separation means each bird must steer to avoid bumping into each other or any other object in the environment. Alignment keeps the each individuals moving in a similar path by taking the average heading of local flockmates. Cohesion keeps the flock together as each bird steers toward the average position of local flockmates. Blackwell has employed a similar algorithm to parameterise a granular synthesis engine, creating an eerily 'lifelike' movement of sound swarming through time.

5.3 Integrating the interactive machine

In the Boids algorithm above, note that the future position of each agent is described in terms of the current state of the other agents: the agents are sensitive to, and respond to changes in their environment. This is obvious when we consider what happens to a real, or simulated flock when it encounters an obstacle: the flock will part to avoid it before rejoining. As well as mapping the behaviour of the flock into musical space then, we can apply the mapping in reverse so that sound events (created by 'live' musicians) in the real world can be mapped into the virtual world to influence the behaviour of the flock. This provides a novel and interesting mechanism for interaction which extends the classic approaches to interactive music.

Traditional approaches to interactive music are based on models of interaction derived from existing musical practices, either allowing performers to control aspects of a predetermined score (eg Machover, 1991) mimicking interpersonal relations in performance (eg Winkler 1991), or extending the performer's relation with their instrument (Machover and Chung 1989). The interesting thing about the use of adaptive systems in an interactive context is that the system amalgamates the characteristics of all these categories, creating at once a responsive composition and a dynamic, behaving instrument which in performance can feel like another, albeit digital, performance partner. Other practitioners have been exploring adaptive models in improvised performance (eg Eldridge 2005, Bown and Lexer 2006) and performances made with systems such as homeostatic networks and continuous time recurrent neural networks demonstrate the success of the approach in integrating an experimental machine aesthetic within the rich (and palatable!) traditions of live performance.

6 Concluding Remarks

As well as providing a short introduction to evolutionary computing techniques, this chapter has described a variety of ways in which they can be used in musical settings, highlighting a number of important issues that arise. At a technical level it is crucial to appreciate the significance of appropriately designed components for any EA-based system, but in an artistic endeavour such as musical composition it is also imperative to engage fully with the wider community:

> "In order to assess the usefulness of [evolutionary computing] in musically creative tasks however, more general discussion of the musical output needs to be conducted. It needs to be recognized that the task is not simply one of computer science, but must include discussion in the relevant domain. This will require the skills and engagement of the wider musical academic community, and an increased number of interdisciplinary research projects." Gartland-Jones and Copley (2003), p.54.

Keeping the composer and/or performer firmly in the loop is one way to help encourage this. The development of tools to allow composers to sympathetically

exploit appropriate properties of adaptive algorithms, and the integration of adaptive systems within live performances, are very promising directions.

It is early days yet, but there is significant potential for exciting and fruitful developments of evolutionary and adaptive computing in music.

Acknowledgements
This paper is dedicated to the memory of our late friend and colleague Drew Gartland-Jones who was crucial to the development of much of the thinking presented here.

References

Back, T., Hoffmeister, F. and Schwefel, H. (1991) A Survey of Evolution Strategies. In R. Belew and L. Booker (Eds), *Proc. 4th Int. Conf. on GAs*, 2-9,Morgan Kaufmann.

Back, T. and H.-P. Schwefel (1993). An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation* **1(1)**: 1-23.

Bagley, J. (1967) *The behaviour of adaptive systems that employ genetic and correlation algorithms*. PhD Thesis, Univ. Michigan.

Beer, R.D. (1990) *Intelligence as Adaptive Behaviour: An Experiment in Computational Neuroethology*, Academic Press.

Bentley, PJ (Ed.) (1999). *Evolutionary Design by Computers*. Academic Press, London.

Biles, J. (1994)'Genjam: a genetic algorithm for generating jazz solos' in *Proceedings of the International Computer Music Conference, 131-137. Aarhus, Denmark.* Available online at: http://www.it.rit.edu/-jab/Genjam94/Paper.html

E. Bilotta, E. R. Miranda, P. Pantano and P. Todd (Eds.) *Proceedings ALMMA 2001: Artificial Life Models for Musical Applications Workshop, ECAL 2001*, Editoriale Bios.

Blackwell, T. and Young, M. (2004) Self-organised Music. *Organised Sound* **9**(2): 137-150.

Bown, O and Lexer, S. (2006) Continuous-Time Recurrent Neural Networks for Generative and Interactive Musical Performance. In *Proc. Evomusart Workshop 2006* (Forthcoming)

Burton, A.R. and Vladimirova, T. (1999) 'Generation of musical sequences with genetic techniques' *Computer Music Journal* **23(4)**, 59-73.
Collins, R. and Jefferson, D. (1991) Selection in massively parallel genetic algorithms. In Belew, R. and Booker, L. (Eds), *Proc. 4th Int. Conf. on GAs*, 249-256, Morgan Kaufmann.

Cook, N. (1990) *Music, Imagination, and Culture*. Oxford: Clarendon Press.

Cope, D.(2000) *The Algorithmic Composer* A-R Editions Inc., Wisconsin.

Dahlstedt, P. (2001), Creating and Exploring Huge Parameter Spaces: Interactive Evolution as a Tool for Sound Generation, Proceedings of International Computer Music Conference, Habana, Cuba.

Eiben, A.E. and J.E. Smith, *Introduction to Evolutionary Computing*, Springer, 2003.

Eldridge, A.C. (2005) Cyborg dancing: generative systems for man machine musical improvisation. In *Proceedings of Third Iteration*, Melbourne, Australia

Fisher, R. A. *The Genetical Theory of Natural Selection*, Oxford: Clarendon Press, 1930.

Darwin, C. (1859) *The Origin of Species*, London: John Murray.

Davis, L. (1990) *The Handbook of Genetic Algorithms*. Van Nostrand Reinhold.

Fogel, L.J., A. J. Owens, and M. J. Walsh. Artificial Intelligence through Simulated Evolution. New York: John Wiley, 1966.

Garcia, R. (2001). Growing sound synthesizers using evolutionary methods. In E. Bilotta et al.

Gartland-Jones, A. (2003) Music Box: a real-time algorithmic composition system incorporating a distributed interactive genetic algorithm. In G. Raidl et al. (Eds), *Proc. EvoWorkshops/EuroGP 2003*, 490-501, Springer.

Gartland-Jones, A. and Copley, P (2003) The Suitability of Genetic Algorithms for Musical Composition *Contemporary Music Review*, **22(3)**, 43-55.

Gartland-Jones, A. and Copley, P (2005) Musical Form and Algorithmic Solutions in *Proceedings of the Creativity & Cognition Conference, Goldsmiths College, London*, 226-23, ACM.

Garnett, Guy E. (2001) The Aesthetics of Interactive Computer Music *Computer Music Journal*: 25(1). Cambridge, MA: MIT Press: 21-33.

Goldberg, D. (1989) *Genetic Algorithms*. Addison-Wesley.

Grefenstette, J. (Ed) (1987). *Proc. 2nd Int. Conf. on GAs*. Lawrence Erlbaum.

Haldane, J. B. S. *The Causes of Evolution*, London: Longman, Green and Co., 1932.
Harvey, I. (1992) Species adaptation genetic algorithms: a basis for a continuing SAGA. In F.J. Varela and P. Bourgine (Eds.)*Proc. 1st European Conf. on Artificial Life*. MIT Press/Bradford Books, Cambridge, MA, 346-354.

Hillis, W.D. (1990) Co-Evolving parasites improve simulated evolution as an optimization procedure. *Physica D*, **42**, pp.228-234.

Hodgson, P. (1999) Modelling cognition in musical improvisation through evolution. In A. Patrizio, GA Wiggins, and H. Pain (Eds), *Proc. AISB'99 Symposium on Musical Creativity*, 15-19, SSAISB.

Hodgson, P. (2002) Artificial evolution, music and methodology. In *Proc. 7th Int. Conf. on Music Perception and Cognition*, Sydney, 244-248.

Holland, J. (1962) Outline for a logical theory of adaptive systems. *Journal of ACM*, **3**.

Holland, J. (1966) Universal Spaces: A basis for studies of adaptation. In E.Caianiello (Ed), *Automata Theory*, Academic Press, 218-231.

Holland,J. (1975) *Adaptation in Natural and Artificial Systems.* University of Michigan Press.

Horner, A. and Goldberg, D. (1991) Genetic algorithms and computer assisted music composition. In R. Belew and L. Booker (Eds), *Proc. 4th Int. Conf. on GAs*, 437-441, Morgan Kaufmann.

Husbands, P. An Ecosystems Model for Integrated Production Planning. *Int. Journal of Computer Integrated Manufacturing* **6(1&2)**, 74--86, 1993.

Huxley, J. S. *Evolution: The Modern Synthesis*, London: Allen and Unwin, 1942.

Jacob, B (1996) Algorithmic composition as a model of creativity, *Organised Sound* **1(3)**, 157-165.

Jakobi, N. (1998). Evolutionary robotics and the radical envelope of noise hypothesis. *Adaptive Behaviour*, **6**:325-368.

Johnson, C. (1999). Exploring the sound-space of synthesis algorithms using in interactive genetic algorithms. In A. Patrizio and G. Wiggins and H. Pain (Eds.), Proc. AISB'99 symposium on AI and musical creativity. Brighton: SSAISB.

Juillé, H. and Pollack, J.B. (1996). Co-evolving intertwined spirals. In *Proceedings of the Fifth Annual Conference on Evolutionary Programming*, 461-468, MIT Press.

Keller, H.(1987) *Criticism* London: Faber & Faber.

Koza, J.R. (1992), *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press.

Langton, . (1995) *Artificial Life: An Overview*, MIT Press.

Lischka, C. (1991) Understanding Music Cognition: A Connectionist view. In G. De Poli and A. Piccialli and C. Roads (Eds), *Representations of Musical Signals*, MIT Press, 417-445.

Machover, T., and Chung, J. (1989) Hyperinstruments: Musically Intelligent and Interactive Performance and Creativity Systems. *Proceedings of the 1989 International Computer Music Conference.* San Francisco: International Computer Music Association, 186-190.

Machover, T. (1991) Program notes for the International Computer music conference. San Fransisco: International computer music association

Mandelis, J. (2001), Genophone: An Evolutionary Approach to Sound Synthesis and Performance, In E. Bilotta et al, 37-50.
http://www.cogs.susx.ac.uk/users/jamesm/Papers/ECAL(2001)ALMMAMandelis.ps

Mandelis, J. (2002), "Adaptive Hyperinstruments: Applying Evolutionary Techniques to Sound Synthesis and Performance," Proceedings NIME 2002: New Interfaces for Musical Expression, Dublin, Ireland, pp. 192-193.
http://www.cogs.susx.ac.uk/users/jamesm/Papers/NIME(2002)Mandelis.pdf

Mandelis, J. and Husbands, P. (2003) Musical Interaction with Artificial Life Forms: Sound Synthesis and Performance Mappings. *Contemporary Music Review*, **22**(3):69-77.

McCormack, J. (2003) Evolving sonic ecosystems. *Kybernetes: The International Journal of Systems & Cybernetics*, **32**(1/2), 184–202.

Michalewicz, Z. and Fogel, D.B. (2004) *How to Solve It: Modern Heuristics*, Springer.

Miranda, E. R. (2000). "The Art of Rendering Sounds from Emergent Behaviour: Cellular Automata Granular Synthesis", *Proceedings of 26th EUROMICRO Conference*, Maastricht, The Netherlands. (Published by IEEE Computer Society).

Miranda, E.R. (2003) On the Music of Emergent Behaviour: What Can Evolutionary Computation bring to the Musician ? *Leonardo*, **36**(1), 55-59.

Mitchell, M. (1996) *An Introduction to Genetic Algorithms*, MIT Press.

Morris, R.O. (1922) *Contrapuntal Technique In The Sixteenth Century*, Oxford University Press.

Mulder, A. (1994) Virtual musical instruments: accessing the sound synthesis universe as a performer. In *Proceedings of the First Brazilian Symposium on Computer Music*, 243–250. Caxambu, Brazil.

Pollack, J., M. Bedau, P. Husbands, T. Ikegami and R. Watson (Eds), *Artificial Life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*, MIT Press, 2004.

Pressing, J. (1990), Cybernetic issues in interactive performance systems, *Computer Music Journal*, 14 (1), 12-25.

Rechenberg, I. (1965) Cybernetic solution path of an experimental problem. Royal Aircraft Establishment Translation No. 1122, Ministry of Aviation, Farnborough.

Reynolds, C. W. (1987) Flocks, Herds, and Schools: A Distributed Behavioral Model, *Computer Graphics*, **21**(4) ,25-34.

Rosen, C. (1980) *Sonata Forms*, New York: Norton.

Rovan, J.B. Wanderley, M.M. Dubnov, S. & Depalle, P. (1997), Instrumental Gestural Mapping Strategies as Expressivity Determinants in Computer Music Performance, presented at "Kansei - The Technology of Emotion" workshop.

Schaal, S., A. Ijspeert, A. Billard, S. Vijayakumar, J. Hallam and J.-A. Meyer (2004) *From animals to animats 8: Proceedings of the Eighth International Conference on the Simulation of Adaptive Behavior*, MIT Press.

Schmitt, L.M. (2001), Theory of Genetic Algorithms, *Theoretical Computer Science* 259, 1-61.

Schoenberg, A. (1967) *Fundamentals of Musical Composition*, Gerald Strang (Ed) London: Faber & Faber.

Sims, K. (1991) Artificial Evolution for Computer Graphics, Proc. Siggraph '91, 319-328.

Sims, K. (1994) Evolving 3D Morphology and Behavior by Competition, In R. Brooks and P. Maes (Eds), *Proc. Artificial Life IV*, MIT Press, 28-39.

Singh, R.S. and C. B. Krimbas (2000) Evolutionary Genetics: From Molecules to Morphology, CUP.

Straus, J.N. (1990) *Remaking the Past.* Cambridge, Massachusetts: Harvard Press.

Todd, S. and Latham, W. (1992) *Evolutionary Art and Computers.* Academic Press.

Turing, A.M. (1950). Computing machinery and intelligence, *Mind* LIX 236, 433-460

Vose, M.D. (1999), *The Simple Genetic Algorithm: Foundations and Theory*, MIT Press.

Werner, G.M., & Todd, P.M. (1997) Too many love songs: Sexual selection and the evolution of communication. In P. Husbands & I. Harvey (Eds.), *Fourth European Conference on Artificial Life* (pp. 434-443). Cambridge, MA: MIT Press/Bradford Books.

Werner, G. and Todd, P (1998) Frankensteinian methods for evolutionary music composition. In Niall Griffith and Peter Todd (Eds) *Musical Networks: Parallel Distributed Perception and Performance,* 313-339. Cambridge, MA. MIT Press / Bradford Books.

Wessel, D. and Wright, M. (2000) Problems and prospects for intimate musical control of computers.*Computer Music Journal* **26**(3), 11–22.

Wiggins, G, Papadopoulos, G, Phon-Amnuaisuk, S. and Tuson, A. (1998) Evolutionary methods for musical composition in *Proceedings of the CASYS98 Workshop on Anticipation, Music and Cognition. Liege, Belgium.*Available online at**:** http://www.soi.city.ac.uk/-geraint/papers/CASYS98a.pdf

Winkler, T. (1991) Interactive Signal Processing for Acoustic Instruments. In *Proceedings for the 1991 International Computer Music Conference.* San Francisco, CA: Computer Music Association.

Woolf, S. (1999), Sound Gallery: An Interactive Artificial Life Artwork, MSc Thesis, School of Cognitive and Computing Sciences, University of Sussex, UK.

Wright, A.H.; Vose, M.D.; De Jong, K.A.; Schmitt, L.M. (Eds.) (2005) *Foundations of Genetic Algorithms: 8th International Workshop, FOGA 2005,* Lecture Notes in Computer Science, Vol. 3469, Springer.

Yee-King, M. (2000).,AudioServe - an online system to evolve modular audio synthesis circuits, *MSc Thesis,* School of Cognitive and Computing Sciences, University of Sussex, UK.