16

Evolutionary Optimisation of Mechanical Structures or Systems

Marcelin Jean-Luc

1. Introduction: the need for an integrated optimal design process

The research of the best compromise between economic, mechanical and technological imperatives has always been the primary objective of the mechanical engineer. The methods used to achieve these excellence objectives have evolved considerably over the last few years. The author's experience in optimisation began in 1983. At this time, the design stage would come first, then the calculation and finally optimisation afterwards. In practice, and during experience of shape optimisation of mechanical structures, between 1985 and 1990, many extreme cases were encountered. In these cases, the question of optimisation wasn't posed until damage had occurred in service; the author's industrial partners realized, often too late, that their designing left quite a bit to be desired. They would then call for the author's help in using optimisation programs to supply them with an improved shape. These shapes were reached despite technological limitations being very severe at this stage; so severe, in fact, that engineers were powerless to resolve the problem. Innumerable problems such as this were dealt with. Figure 1 exemplifies this very well. In this case, the very localized optimisation of the rear bearing of a hydraulic hammer is presented (the type of which had been sold in most parts of the world). The bearing in question would break after relatively few cycles of operation. The automatic optimisation of the shape of this product would, simply by a small modification of shape (which would be difficult to predict other than by calculation (increased radius, decreased width), considerably improved the mechanical durability of the bearing: the over-stress being reduced by 50%, the objective being the minimisation of the maximum value of the *Von Mises* equivalent stress along the mobile contour, whilst taking into account the technological constraints of the industrial partners.

Such an approach to designing has become unthinkable these days. The economic competitivity has increased, the design and manufacture delays have

Source: Manufacturing the Future, Concepts - Technologies - Visions , ISBN 3-86611-198-3, pp. 908, ARS/plV, Germany, July 2006, Edited by: Kordic, V.; Lazinica, A. & Merdan, M.

been reduced and therefore the numerous overlaps that this approach involves have become prohibitive. In short, optimisation can no longer be separated from the act of designing. It is now admitted that in an integrated design approach, optimisation has to begin from the design stage taking into consideration the constraints both of specification and those induced by different materials. Optimisation is therefore made easier because constraints or limitations can be more easily varied, in accord with all those involved with the project. Examples will be found in (Clozel, 1991), (Guillot et al., 1989), (Hernot et al., 1995). This was not the case in the preceding example, where the optimisation did not take place until after being put into service, and which became an extremely constrained problem.

In this chapter, it will be shown that the integration of optimisation from the design phase is, according to the author, possibly thanks to the new optimisation techniques. A certain number of optimisation methods are popular at the moment, which are known as probabilistic or stochastic. For example, the simulated annealing method or genetic algorithms, whose principle advantages are assured convergence without using derivatives and eventual functions with discrete and non-derivable variables, even though determinist methods of optimisation (called gradient methods) necessitate a calculation resistant to these sensitivities.

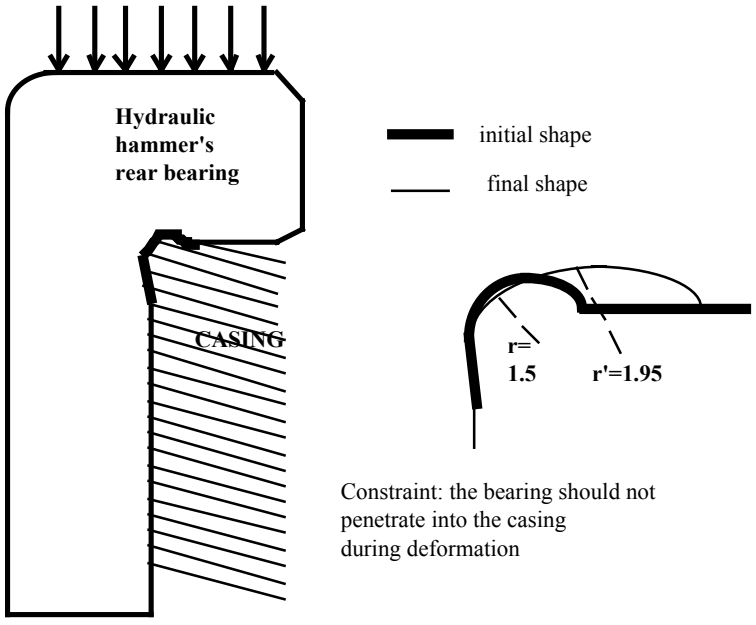

Figure 1. Optimisation of the shape of a hydraulic hammer's rear bearing

Genetic algorithms rely on the natural laws of selection which allow a living organism to adapt to a given environment. From these principles, it seems sensible to apply genetic algorithms to the optimisation of mechanical structures. As will be shown in precise examples, genetic algorithms will allow, from the beginning of the design process, adaption of the mechanical object to it's environment and to the specifications. It will be seen, especially on the example of a stiffened plate (part 3.3), how a product responding to the specifications of stiffness, weight, etc..., can be obtained directly.

After a presentation of the methods and tools used (in part 2), this chapter focuses on applications entering into the field of mechanical technology and the analysis of mechanical systems and processes (part 3). It will be seen in the conclusion (part 4), that the difficulties are more important in the case of an integrated, optimal design process of mechanical systems, because of the complexity of the problems. Nevertheless, it will be seen in this conclusion that integrated optimisation and even alternatives to A.I. (artificial intelligence) techniques can effectively be considered, for precise problems of mechanical technology, such as the optimisation of gears (part 3.1) or the construction of a mechanism (part 3.2). The conclusion supplies possible solutions for the problem in its entirety.

2. The methods used: optimisation tools adapted to mechanical technology

In addition to what has already been mentioned, the author's experience began with the shape optimisation of mechanical structures (2-D and symmetrical), although this was in the context of traditional design. See (Trompette & Marcelin, 1987), (Marcelin & Trompette, 1986), (Marcelin & Trompette, 1988), (Steffen & Marcelin, 1988).
Mathematical optimisation programs were quite difficult to use and not sufficiently versatile to be adapted quickly to new cases. In the opinion of the author, the optimal integrated design could not be achieved with normal mathematical programming techniques, which require a formulation heavily adapted to each particular problem. It will be shown in this book, that stochastic techniques are ideally suited to integrated optimisation and to mechanical technology problems.

Note that the essential characteristics of the problems are as follows:

- the design variables are often a mixture of discrete and continuous values;
- they are often highly constrained by strict technological constraints.

The problem is to maximise a function of n variables. The principle of genetic algorithms is to make a population of individuals evolve according to a replica of Darwinian theories. The starting point is a population of individuals chosen randomly and coded by binary numbers (as an example) called chromosomes. From this point, the algorithm generates, more or less randomly, new populations formed from individuals, increasingly more adapted to a given, well-defined environment. Selections and reproductions are made from the best performing parents of the population from which they come. They are stochastic or deterministic. The creation of these offspring is done by the application of genetic operators (mutation, crossing). It is always stochastic. The new replacement population is created by the selection of the best performing individuals, among either the offspring or the parents of the offspring. The replacement is either stochastic or deterministic. In the books (Goldberg, 1989), (Koza, 1992), (Michalewicz, 1996), (Rumelhart & McClelland, 1986), additional information can be found along with a demonstration of the convergence of the method.

The essential advantage of these methods is that they operate simultaneously on a test space of the solutions. The genetic method differs from the simulated annealing method by the operators which are used to force the evolution of the test population. In all cases, the convergence is always assured towards an extreme. This extreme is not necessarily the absolute extreme, but has more chance of being so, than if a traditional gradient method is used. This is shown in (Goldberg, 1989). In effect, a stochastic method explores a larger solution space. In addition, another essential advantage of these methods lies in the small number of assumptions that are required for the objective function.

2.1 Genetic algorithms

The genetic algorithms used are optimisation algorithms and make up part of the stochastic methods. They were first used in 1975. As their name implies, these algorithms seek the optimal solutions to a given problem by simulating the evolution and adaption of living organisms.
"The individual most able to adapt to a well defined environment has the greatest chance of continuing to survive and transmitting it's qualities to new individuals."

When Charles Darwin claimed his theory of the "Survival of the Fittest", a new century of understanding nature and life began. Over millions of years, life on earth has been in a process of optimal adaption to current environments. A natural selection between different individuals has maintained a changing of their genetics, that way the fittest ones, in the sense being best adapted, have survived and transferred their genetic codes to their descendants by a randomised information exchange. On the other hand, the worst adapted ones have died off.

This process is a process of optimisation. The natural selection is like a search algorithm for finding the best solution of living in a particular, natural environment. Certainly, the system of nature can not easily be transferred to technical systems. But we can have a look at, how the main rules of selection in nature are processed. And we can try to abstract and implement these for solving problems of optimisation with the help of computers. David Goldberg (Goldberg, 1989) described, based on John Holland's work (Holland, 1975), two important subjects:

- explanation of the adaptative processes of life and other natural systems,
- design of artificial systems which behave similarly like natural systems concerning their important mechanisms.

As biological systems obtain such a robustness, efficiency and flexibility, the basic rules of their mechanisms have been very interesting for artificial systems of engineering, computer or business applications. A genetic algorithm, now shortly called G.A., is a stochastic search method with the aim to scan randomly through these areas of the search space where the biggest chance of success seems to be provided. Today, G.As are proved theoretically and empirically for searches in complex spaces. They are simple for computer implementation but very powerful and effective. And there are no fundamental restrictions or limitations about the search space like continuity, existence of derivatives, unimodality, and so on.

The process of optimisation is a performance towards an optimum. This means that there is, on the one hand, the process of improvement and, on the other hand, the reaching of the optimum. Mostly, not only destination of the maximum is important, as supported by many conventional methods, but also a comparison of being or behaving better relatively to others, like the G.A. do this. The following points are figuring out the differences between G.As and other methods:

- as G.As work with a special coding of the parameter set, they can use particular similarities of this coding in a very general way. Conventional methods use the parameters themselves, and often they are restricted by a lot of limitations (continuity, unimodality, ...),
- most algorithms seek point-by-point to find the peaks in the search space. G.As do this in a quasi paralell way with a population of many strings,
- G.As have no use of any auxiliary functions like the derivatives. They use only a so-called payoff information of the objective function,
- G.As work with probabilistic transition rules, not with deterministic rules. But this stochastic effect leads to an improvment in searching, not to a random search.

Conventional search methods are not very robust, but often specially designed for particular problems, they are mostly succesful in many applications. Even sometimes, their performance can be better than these of G.As. But nevertheless, already simple G.As have their own advantages. To apply this process to optimisation, the starting point is a group of solutions to the posed problem. This group is called the initial population, and is chosen randomly from the valid domain. Each individual solution is called a chromosome, which carries information relevant to evaluate the cost function of each chromosome. This information is encoded in symbols (usually natural numbers) of which each symbol is a gene. It is known that each chromosome has the same number of genes. The greater the size of the population, the higher the effectiveness of the search, as the solution space will be explored in a wider manner. It is always limited, however, by the time needed for calculation and the capacity of the operating system. Once the initial population is generated, the algorithm begins to create a new population, equal in size to the initial population, in the case of a simple G.A.. The individuals of the new population will be developed from those of the preceding population, after having been subjected to a number of operators. In this way, each population generates a new one. The algorithm stops after a given number of generations, or by fixing other relevant function-cost criteria. In living organisms, the genetic operators are applied to the chromosomes of the parents; by analogy, the operators used in genetic algorithms are applied to the different codings of the individuals of a generation. In general, the genes can take binary values (0 or 1) to represent the state of a given piece of information. In particular cases, a gene can take a certain number of fixed values.

Genetic algorithms make up a large family of algorithms, each one different

from the other by their degree of simulation of different phenomena, related to the adaption to natural systems.
By limiting to the application of a simple G.A., the three operators which are going to be used will be:

- reproduction
- crossing
- mutation

The main idea of these operators comes from the observation of the interaction in natural surroundings. The problem is always translated into terms of maximising the objective function. The greater the objective function value of an individual relative to other individuals of the same population, the greater its chances of selection. The selection is carried out randomly, respecting the weighting of each individual.

$$P_{sel\ i} = \frac{f_i}{\Sigma_{pop} f}$$

The selection operation is applied as many times as are necessary to complete the population.

Once the selection is finished, the individuals are paired, and a crossing operator is applied to each. The probability of crossing is Pc, which is fixed beforehand. This operator consists of choosing a random position for the two chromosomes, cutting them at that position and changing the two parts beyond that position.

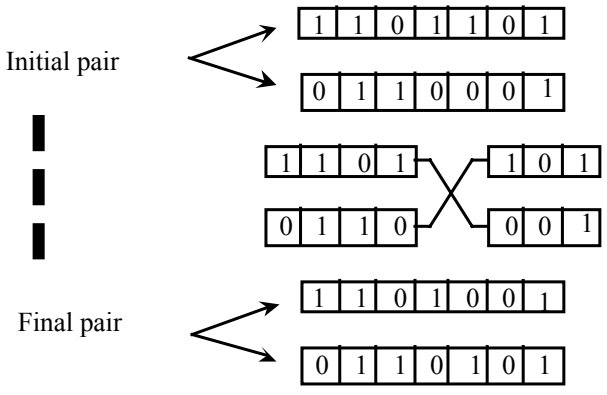

Figure 2. Crossing diagram

Reproduction and crossing are operators which transmit the good qualities of one generation to another. Important information, however, might not be represented by the chromosomes of the mother generation, or those carrying it might be destroyed accidentally by the transmission operators. Mutation consists of changing the values of a certain number of genes chosen randomly from those carried by the whole population. The probability Pm of applying mutation to a gene is fixed beforehand. If a gene is chosen to undergo a mutation, the new value is chosen randomly from among all the possible values which it could take.

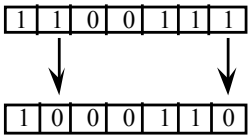

Figure 3. Mutation diagram

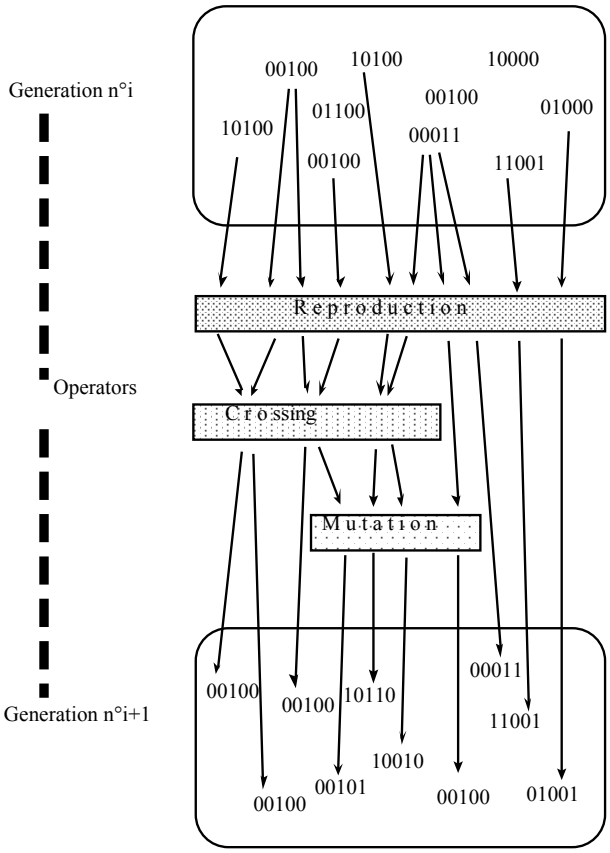

Figure 4. Diagramatic representation of the simple G.A.

The operators mentioned apply themselves to a generation to create another, respecting the sequences already stated. Figure 4 gives an overview of the actions of the three operators on a generation. The genes can take three values: 0, 1 or 2.

2.2 The simulated annealing method

The simulated annealing method, as the name implies, is based upon the metallurgical process of the same name. This "random search" method of minimisation is characterized by accepting the increases of the objective function with a given probability. This allows it to get out of the troughs (unlike deterministic methods) and therefore to escape from local minima. In the metallurgical process of annealing, if a metallic body is heated to it's melting point and then slowly cooled to ambient temperature, then the global energy of the metal will eventually pass through an absolute minimum. The basic algorithm is the "metropolis" algorithm, which is the standard of random research methods. Here is a reminder of this very simple algorithm:

Step 1: choose initial value of X0, evaluate F(X0), for k=0

Step 2: at the k+1 iteration, create a vector X, from Xk; if F(X)<F(Xk) then Xk+1=X, else Xk+1=Xk.

Step 3: if the finishing criteria have not been met, then let k=k+1 and go to step 2. If the criteria have been met then finish.

Several theorems of convergence to a global minimum were established for these methods. Difficulties in escaping from local minima remained however, which is why, of course, simulated annealing is needed.
The metallurgical process of annealing is applied to the optimisation problem. The objective function, F, is equivalent to an energy term. A temperature function, T(k), is introduced, whose purpose is to allow acceptance of the growth, by using the probability: p = exp(-DELTA(F)/T(k)). The principles and details were given in (Bonnemoy & Hamma, 1991). We shall see an application of this method in part 3.2.

2.3 Neural networks

The operation of artificial neural networks, as their name suggests, takes inspiration from that of biological neural networks. A large part of their vocabulary

is therefore been borrowed to describe them. In the author's opinion, this is as far as the similarities go. Detail of the theory, which will be summarized later, can be found in (Jodouin, 1994). The use of neural networks for the simulation or modelisation will be done in two stages: one phase which is called apprenticeship, using finite elements calculations for example in mechanics of structures, followed by a calculation or generalisation phase. In the present case, neural networks should be able to estimate an objective function or a cost function of entry or design variables. It should be noted here that the entry variables will be the binary digits of the chromosomes when using a G.A. or the real values of the design variables when using the simulated annealing method. To describe a neural network, it is sufficient to know the neuron model and the arrangement of the connections between the neurons.

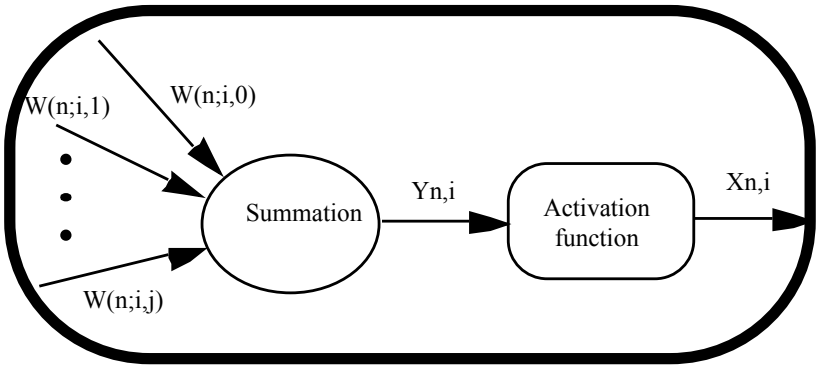

Figure 5. The elementary neuron

A neuron is modeled by two operators (figure 5). Firstly, a summing operator which develops a potential Yn,i, equal to the balanced sum of the cell entries (it is this which will be translated by the optimisation of balanced weights in the apprenticeship phase). Secondly, an operator which calculates the state of the exit value Xn,i = f(Yn,i) of the neuron as a function of its potential (f is called the neuron function, it can be either linear or non-linear). The entries are the exit values of the same layer or of another layer, or eventually the exterior entries themselves.

In the case of the optimisation procedure and binary coding for the chromosomes of the G.A., the exterior entries will be 0s or 1s, which will correspond to the chromosome digits. The function, f, can take different forms depending on

the type of network (figure 6). The most up to date models of connection networks are defined in figure 7. A complex network can be split into many layers and in this sequence of layers, a direction of information transfer can be defined. Furthermore, in the interior of one layer or between two layers, the connections can be partial or total.

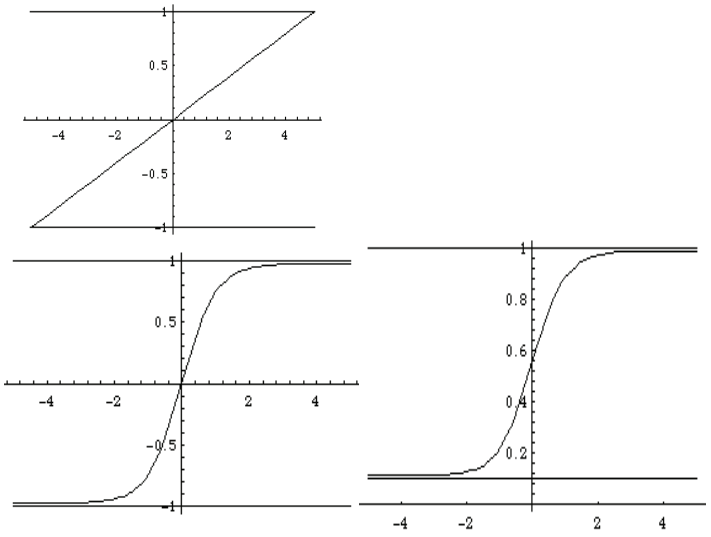

Figure 6. Some neural functions

The apprenticeship phase consists of optimising or adapting, by an apprenticeship rule, modifying the weights at each link. An apprenticeship sample is used to do this, that is, solutions which will be previously determined by finite element analysis for example in mechanics of structures. The principle criterion is to have a minimal error for the evaluation of the function. Local adaption rules (for which the weight optimisation is based on the states of the neurons connected to corresponding links) are distinguished from other rules which are much more difficult to put into use. Among the rules which are called local, and for which details can be found in (Jodouin, 1994), the best known are those which are called supervised or non-supervised, and the iterative rules. Among the non-local rules, are two of the most frequently used. Firstly, there is the Widrow-Hoff rule. This applies in particular to completely interconnected two layer networks. This was generalized in multi-layer networks by the retropropogation algorithm, to the error gradient. The error at the exit of each neuron being expressed as a function of the error calculated for the following layer by a simple differential calculation. Secondly, the Hopfield model

for the adaption rule is based on the minimisation of total energy of the network, which is the sum of the elemental energies which characterize a neuron. Each neuron is updated in a randomly drawn series.

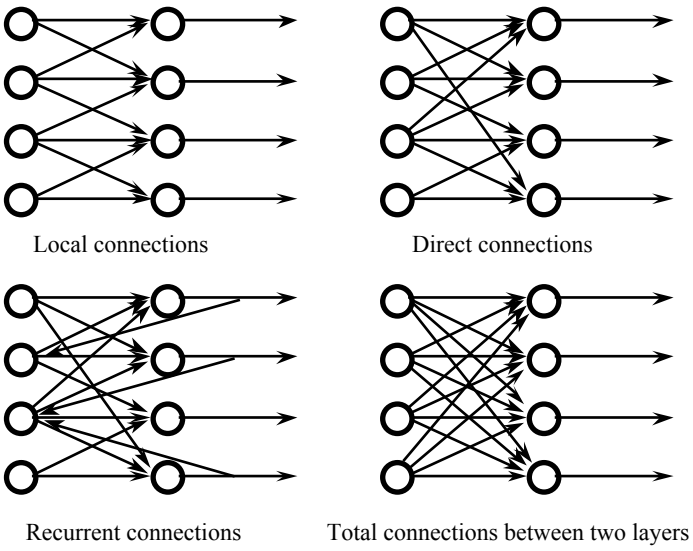

Figure 7. Some models of network connections

To summarize, a neural network works in two phases. In the first place, the apprenticeship phase, during which the adaption function is active. This allows the weight values to be optimised from a set of entry values (the design or conception variables) and from exit values (the objective function(s) or cost function) called the apprenticeship set. In the second place, the calculation or generation mode, during which the values of the weights are fixed. This allows the calculation by the neural network of the exit values as a function of the entry values.

The application of neural networks to modelisation, especially for the simulation of the calculations for the mechanical structures, seems promising from the results obtained. See (Berke & Hajela, 1992), (Szewczyk & Hajela, 1994) and (Hajela & Szewczyk, 1994). The continuation to modelisation seems natural as the action of modeling a process or a behavior, necessitates the knowledge of the principle characteristics of the process or behavior. The network knows how to extract these characteristics and can therefore be memorised easily. On the other hand, this ability to model exploits the adaption qualities of networks, allowing them to improve as they are exploited. In this work, effective

neural networks were used, at the current level of knowledge, and for which apprenticeship and generalisation/calculation algorithms are described in (Jodouin, 1994). This neural network, quite easily programmed, is a three layer network with a sigmoid neural function (figures 6 and 8) called MLP (multilayer perceptron). The MLP has been used in most of our applications.

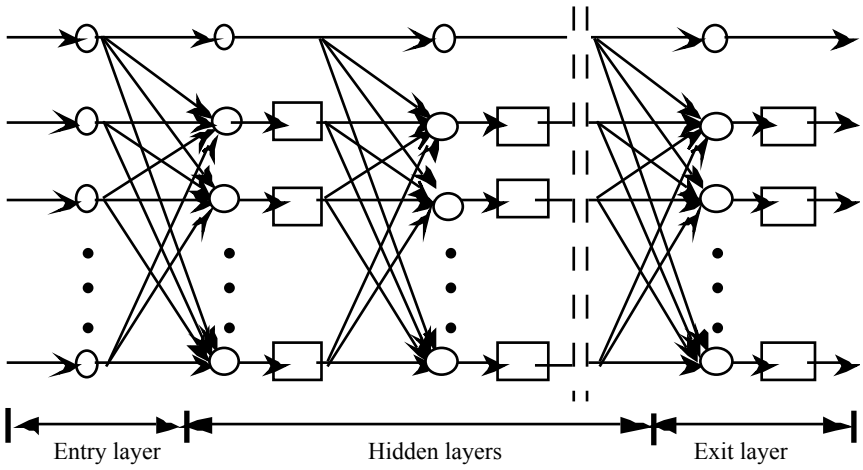

Figure 8. Neural networks used

3. Integrated optimal design of particular mechanical systems and process

3.1 Optimisation of gears

Gears are very complicated components. A large number of dominating factors vary in every case: radius of curvature, unitary loading, pressure, slip speeds, etc.... The design variables are huge in certain cases and take very discrete values (such as the module, the choice of materials). Often several objectives work in competition: balancing the energy transmission in bending and under pressure, optimisation of masses, balancing the slips, to mention but a few. The idea consists of automatically dimensionning a right-sided cylindrical gear or helical gear, so as to find a good compromise between a minimum weight, dynamic performance (energy transmission) and geometric criteria such as balancing the slips. This optimisation problem is very difficult to re-

solve by hand and often leads to compromised solutions that are not entirely satisfactory, so therefore the idea of an automatic optimisation technique is most desirable for this complex problem. In (Daidie, 1993), the authors of the paper propose a classic optimisation technique for gears. Nevertheless, these mathematical optimisation methods depend on the understanding of objective function gradients and are difficult to adapt to gears for three principle reasons:

a) initiall, certain design variables are continuous while others are discrete,
b) the derived programming is quite delicate because the optimising functions often depend implicitly on the design variables,
c) finall, the major flaw is that these methods become blocked at a local extreme (often the only way to pursue the program is to rerun the calculation with a new starting point). Thus all specialists know, the optimisation of gears is acknowledged to have numerous solutions and often it is better to adapt them to given situations. This therefore leads to the use of a genetic algorithm in order to solve the problem. Note that in (Mekhilef & Dupinet, 1993) the researchers use a method of simulated annealing (part 2.2) to solve the problem, and with some success. The problem of gear optimisation is illustrated in figure 9.

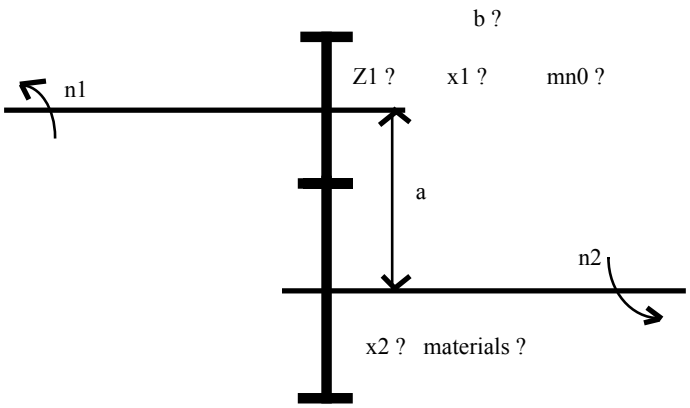

Figure 9. Definition of optimising gearing

There are two main difficulties with this problem. First of all, it is a matter of coding the solution in the form of a chromosome that will be simple and efficient; then there is the matter of finding a good compromise in function of different objectives between the different criteria (weight, power differential, balancing slips).

The coded parameters are restrained so the field of study is not too large and so therefore the chromosome is not too long. So in this way we have not considered all the design parameters in gearing, but only six main parameters: k , z1 , x1 , x2 , m_{n0}, and the material; other parameters such as the helix angle for example are fixed during the course of the optimisation. So in which case, this choice can be modified without any problems (figure 9).

When considering the choice of the objective function, we are using a multi-objective technique where the objective function will infact be a balanced sum of the different functions that we want to obtain, such as for example the minimum weight and minimum difference between slips. In which case we must weight certain objectives in respect to others; the effective choice can easily be reset, in the case of the same script of the different objectives (the shape under which they appear is at the discretion of the researcher in the domaine of optimisation). Above all though, the difficulty consisted of choosing the weighting of the coefficients in respect to their influence (that are of a different nature). This can only be done resulting from numeric experiments, where the goal was to find the best compromise possible between the different objectives. In the first place, the coding of the variables that we have used in the genetic algorithm is as follows: each of the values: k, z1, x1, x2, m_{n0} and materials are written into a binary numeration system. So, six chains are obtained C1, C2, C3, C4, C5, C6, with lengths of 4, 6, 6, 7, 4, 3 respectively.
An example of a genetic identity card (chromosome) for a gearing system is given here.

Genetic identity coding (chromosome) for gearing:

1001 110101 101100 1010100 1001 010
C1 C2 C3 C4 C5 C6

C1 : size coefficient of tooth ‘k’,
C2 : number of teeth ‘Z1’,
C3 : coefficient ‘x1’,
C4 : coefficient ‘x2’,
C5 : real shape module ‘mn0’,
C6 : material chosen from a library of 8 different types.

This coding is limited to a chromosome of a total of thirty genes long which arrange end to end (where the order is not important) the relative information of

the gearing. This coding restrains the admissible domaine of design itself. There are only $2^4 = 16$ possible width 'k' coefficients; only $2^6 = 64$ possibilities for the number of teeth 'z1' (that can vary between 12 and 75 for example); x1 and x2 only vary between -0.5 and 0.5 with two significant numbers; for m_{n0} there are 16 normalised numbers possible; finally, the material for the pinion and gearwheel is the same, and there are eight possible choices from the library of materials. For example the code 001 corresponds to 30CND8, the code 110 to 16NC6, and so on. It therefore follows, that it is possible to modify the structure or the length of the chromosome without too much difficulty.

In the second place, 'multi-objective' functions in the case of gearing are rather complicated. We propose that by the following we can modify at will, in function of the results of diverse numerical experiments. The idea is to build the function as if it were the sum of the weighted representative terms, by the coefficients that we can vary when we wish, more or less according to the importance of such and such a criteria. The function that we have used for the tests that follow, is illustrated below:

$$F=10^{10} - \frac{I_1}{Rap}\left(\frac{b}{b_{max}}\right)\left(\frac{d_1}{d_{1max}}\right)^2 - I_2|g_{s1}-g_{s2}|$$

$$-I_3 \frac{Rap}{Ptrans}\left[\,|P_{rup} - c.P_{trans}|+|P_{pres}-c.P_{trans}|\,\right]$$

I1, I2 and I3: weighting coefficients ,
gs1, gs2: maximum absolute slips,
Rap: ratio of quality against price of material ,
b: width of material,
d1: primitive diameter of pinion.

The presence of the term 10^{10} is due to the fact that the G.A. maximises the functions. To calculate the functions at a minimum, it is possible to look for the maximum of the opposing function plus a very large term. The second term affected by coefficient I1 is a term relating to the minimisation of gear size with relation to a given maximum size. This term is penalized in terms of quality/price of a material. The third term affected by coefficient I2 expresses the equalizing of the absolute slip (very important in reducing wear). Finally, the fourth term, affected by coefficient I3, is a term expressing the search for balance between the transmissible powers under pressure and under flexion, and

also respecting the safety factor with relation to power transmitted. It is possible to add other criteria to this multi-objective function, e.g. a term expressing maximisation of driving relation or a term ensuring imposed distances between axes are respected. After several numeric tests on a basic example, the values of weighting coefficients below were chosen for the following test case: I1=0.2, I2=0.1 and I3=10. This test concerns a helicoidal gear used in a fixed axis gearbox. The parameters of the G.A. are: population size=200 and number of generations=100. The results are compared to a reference solution, optimised using other methods.
The given factors are:

Ptrans=400KW
Nmax(input)=1485 tr/min
transmission relationship u=6
developing circle of teeth ⌀=8°33'
Quality factor Q=6
Life H=200000 hours
functionning with negligible shock.

The following is the best solution of the last generation:

Geometrical analysis :

	mn0	Z1	x1	x2	k	Material
reference solution	5	26	0.44	-0.4	32	16NC6
genetic algorithm	6	23	0.10	0.09	13	16NC6

	reference solution	genetic algorithm
width b (mm)	160	78
d1 (mm)	131.4	139.9
volume bd_1^2	2.7E6	1.5E6
gs1	0.24	0.48
gs2	0.33	0.30
e_e	1.63	1.85
Pflex (kW)	1100	760
Ppres (kW)	1000	740

The objectives have been achieved: a correct balance between absolute slips gs1 and gs2 and powers with a sufficient saftey factor. It is also notable that the